A polynomial time algorithm for computing extinction probabilities of multi-type branching processes

A polynomial time algorithm for computing extinction probabilities of multi-type branching processes*

Kousha Etessami
U. of Edinburgh
kousha@inf.ed.ac.uk

Alistair Stewart
U. of Edinburgh
stewart.al@gmail.com

Mihalis Yannakakis
Columbia U.
mihalis@cs.columbia.edu

Abstract

We show that one can approximate the least fixed point solution for a multivariate system of monotone probabilistic polynomial equations in time polynomial in both the encoding size of the system of equations and in $\log(1/\epsilon)$, where $\epsilon > 0$ is the desired additive error bound of the solution. (The model of computation is the standard Turing machine model.)

We use this result to resolve several open problems regarding the computational complexity of computing key quantities associated with some classic and well studied stochastic processes, including multi-type branching processes and stochastic context-free grammars.

1 Introduction

Some of the central computational problems associated with a number of classic stochastic processes can be rephrased as a problem of computing the non-negative *least fixed point* solution of an associated multivariate system of monotone polynomial equations.

In particular, this is the case for computing the *extinction probabilities* (also called *final probabilities*) for *multi-type branching processes* (BPs), a problem which was first studied in the 1940s by Kolmogorov and Sevastyanov [40] (see, e.g., Harris [35]). Branching processes are a basic stochastic process, with applications in diverse areas ranging from population biology to the physics of nuclear chain reactions (see [35] for the classic theoretical text on BPs, and see [4, 39, 28] for some more recent books). BPs describe the stochastic evolution of a population of objects of distinct types. They are studied both in discrete and continuous time, and in settings with both a finite and infinite number of types. Our focus in this paper is on discrete-time BPs with a finite number of types. In each generation, every object a of each type T gives rise to a (possibly empty) multiset of objects of distinct types in the next generation, according to a given probability distribution on such multisets associated with the type T. The *extinction probability*, q_T, associated with type T is the probability that, starting with exactly one object of type T, the population will eventually become extinct. Computing these probabilities is fundamental to many other kinds of analyses for BPs (see, e.g., [35]). Such probabilities are in general irrational, even when the finite data describing the BP (namely, the probability distributions associated with each of the finitely many types T) are given by rational values (as is assumed usually for computational purposes). Thus, we would like to compute the probabilities approximately to desired precision.

*An extended abstract for this paper appeared in the *Proceedings of the 44th ACM Symposium on Theory of Computing (STOC)*, 2012. Research partially supported by the Royal Society and by NSF Grant CCF-1320654.

Another essentially equivalent problem is that of computing the probability of the language generated by a *stochastic context-free grammar* (SCFG), and more generally its *termination probabilities* (also called the *partition function* of the SCFG). SCFGs are a fundamental model in statistical natural language processing and in biological sequence analysis (see, e.g., [43, 14, 44]). A SCFG provides a probabilistic model for the generation of strings in a language, by associating probabilities to the rules of a CFG. The termination probability of a nonterminal A is the probability that a random derivation of the SCFG starting from A eventually terminates and generates a finite-string; the total probability of the language of a SCFG is simply the termination probability for the start symbol of the SCFG. Computing these termination probabilities is again a key computational problem for the analysis of SCFGs, and is required for computing other quantities, for example the probability of generating a given string.

Despite decades of applied work on BPs and SCFGs, as well as theoretical work on their computational problems, no polynomial time algorithm was known for computing extinction probabilities for BPs, nor for termination probabilities for SCFGs, nor even for approximating them within any nontrivial constant: prior to this work it was not even known whether one can distinguish in P-time the case where the probability is close to 0 from the case where it is close to 1.

We now describe the kinds of nonlinear equations that have to be solved in order to compute the above mentioned probabilities (these equations arise from the *generating functions* for the corresponding BPs [35]). Consider systems of multi-variate polynomial fixed point equations in n variables, with n equations, of the form $x_i = P_i(x),\ i = 1, \ldots, n$ where $x = (x_1, \ldots, x_n)$ denotes the vector of variables, and $P_i(x)$ is a multivariate polynomial in the variables x. We denote the entire system of equations by $x = P(x)$. The system is *monotone* if all the coefficients of the polynomials are nonnegative. It is a *probabilistic polynomial system* (PPS) if in addition the coefficients of each polynomial sum to at most 1.

It is easy to see that a system of probabilistic polynomials $P(x)$ always maps any vector in $[0,1]^n$ to another vector in $[0,1]^n$. It thus follows, by Brouwer's fixed point theorem, that a PPS $x = P(x)$ always has a solution in $[0,1]^n$. In fact, it always has a unique *least* solution, $q^* \in [0,1]^n$, which is coordinate-wise smaller than any other non-negative solution, and which is the *least fixed point* (LFP) of the monotone operator $P : [0,1]^n \to [0,1]^n$ on $[0,1]^n$. The existence of the LFP, q^*, is guaranteed by Tarski's fixed point theorem. From a BP or a SCFG we can construct easily a corresponding probabilistic polynomial system of equations (PPS), $x = P(x)$, whose LFP q^* yields precisely the vector of extinction probabilities for the BP, or termination probabilities for the SCFG. Indeed, the converse also holds: computing the extinction probabilities for a BP (termination probabilities of an SCFG) and computing the LFP of a system of probabilistic polynomial equations are equivalent problems. As we discuss below, some other stochastic models also lead to equivalent problems or to special cases.

Related Work. As already stated, the polynomial-time computabilities of these basic probabilities for multi-type branching processes and SCFGs have been longstanding open problems. In [24], we studied these problems as special sub-cases of a more general class of stochastic processes called *recursive Markov chains* (RMCs), which form a natural model for analysis of probabilistic procedural programs with recursion, and we showed that these problems are equivalent to computing termination probabilities for the special subclass of *1-exit* RMCs (1-RMC). General RMCs are expressively equivalent to the model of *probabilistic pushdown systems* studied in [17, 11]. They also subsume, in a precise sense, a number of other discrete-time stochastic models that arise in queueing theory, including *Quasi-Birth-Death processes* (QBDs) and Tree-like QBDs (see [23]).

We showed in [24] that for the case of BPs, SCFGs, and 1-RMCs, the *qualitative* problem of determining which probabilities are exactly 1 or 0 can be solved in polynomial time in the size of the input (i.e. the number of bits needed to specify the given instance), by exploiting basic results from the theory of branching processes. We proved however that the *decision* problem of determining whether the extinction probability of a BP (or termination probability of a SCFG or a 1-RMC) is $\geq 1/2$ is at least as hard as some longstanding open problems in the complexity of numerical computation, namely, the *square-root sum problem*, and a much more general decision problem (called `PosSLP`) which captures the power of unit-cost exact rational arithmetic [2]. In the unit-cost arithmetic RAM model of computation, all arithmetic operations $(+, -, *, /)$ on rational numbers cost one unit of time, regardless of how long the numbers are, i.e., how many bits are needed to encode them (their numerator and denominator). By contrast, in the standard (Turing) model of computation, the cost of the operations depends on the length (the number of bits) of the numbers. It is strongly believed that unit-cost-P $\neq$ P, where unit-cost-P is the class of decision problems (i.e. problems with Yes/No answer) that can be solved in polynomial time in the unit-cost model, and P is as usual the class of decision problems that can be solved in polynomial time in the standard model. The problem `PosSLP` mentioned above (see section 6 for the definition) is complete for unit-cost-P and thus it is in P if and only if unit-cost-P = P. The fact that the decision problem for the extinction probability of a BP (and termination probability of a SCFG and 1-RMC) is at least as hard as `PosSLP` implies that it is very unlikely that it can be solved in P-time. For general RMCs we showed that in fact this hardness holds for computing *any* nontrivial *approximation* of the termination probabilities, i.e., it is hard to distinguish the case that the termination probability is close to 0 from the case that it is close to 1. No such lower bound was shown for the approximation problem for the subclass of 1-RMCs (and BPs and SCFGs). In terms of upper bounds, the best we knew so far, even for any nontrivial approximation, was that the problem is in the class FIXP (which is contained in PSPACE), i.e., it can be reduced to approximating a Nash equilibrium of a 3-player game [25]. We improve drastically on this in this paper, resolving the problem completely, by showing we can compute these probabilities in P-time to any desired accuracy.

There are a number of natural iterative methods that one can try to use (and which indeed are used in practice) in order to solve the equations arising from BPs and SCFGs. The simplest such method is *value iteration*: starting with the vector $x^0 = 0$, iteratively compute the sequence $x^{i+1} := P(x^i)$, $i = 1, \ldots$. The sequence always converges monotonically to the LFP q^*. Unfortunately, it can be very slow to converge: even for the simple univariate polynomial system $x = (1/2)x^2 + 1/2$, for which $q^* = 1$, one requires 2^{i-3} iterations to exceed $1 - 1/2^{i-1}$, i.e. to get i bits of precision [24].

In [24] we showed a much more rapid method always converges monotonically to q^*. Namely, we showed that a decomposed variant of Newton's method can be applied to such systems of equations (and in fact, much more generally, to any monotone system of polynomial equations) $x = P(x)$, and always converges monotonically to the LFP solution q^* (if a solution exists). Optimized variants of this decomposed Newton's method have by now been implemented in several tools (e.g., [50, 44]), and they perform quite well in practice on many instances.

The theoretical speed of convergence of Newton's method on such monotone (probabilistic) polynomial systems was subsequently studied in greater detail by Esparza, Kiefer, and Luttenberger in [16]. They showed that, even for Newton's method on PPSs, there are instances where exponentially many iterations of Newton's method (even with *exact arithmetic* in each iteration) are required, as a function of the encoding size of the system, in order to converge to within just

one bit of precision of the solution q^*. On the upper bound side, they showed that after *some* number of iterations in an initial phase, thereafter Newton obtains an additional bit of precision per iteration (this is called *linear convergence* in the numerical analysis literature). In the special case where the input system of equations is *strongly connected*, meaning roughly that all variables depend (directly or indirectly) on each other in the system of equations $x = P(x)$, they proved an exponential upper bound on the number of iterations required in the initial phase as a function of input size. For the general case where the input system of equations is not strongly connected, they did not provide any upper bound as a function of the input size. In more recent work, Esparza et al [15] further studied probabilistic polynomial systems. They did not provide any new worst-case upper bounds on the behavior of Newton's method in this case, but they studied a modified method which is in practice more robust numerically, and they also showed that the qualitative problem of determining whether the LFP $q^* = \mathbf{1}$ is decidable in *strongly* polynomial time.

There is also an independent body of recent research which has considered Newton's method, and related iterative methods, applied to equations that arise for extinction of so called *Markovian (binary) trees*, which are essentially equivalent to a subset of multi-type branching processes [31, 6, 32, 33, 30]. These works in the queueing theory and performance evaluation literature, and specifically in the literature on matrix-analytic methods for stochastic processes, by Hautphenne, Latouche, and others, build on a large body of prior work, over several decades, on the numerical analysis of iterative methods for classes of monotone non-linear matrix equations that arise for computing important quantities for various classes of countably-infinite state stochastic processes that are finitely presented as "structured Markov chains", including quasi-birth-death processes (QBDs) and tree-like QBDs (TL-QBDs) (see, e.g., [46, 42, 9, 8]). However, these works in general do not provide any complexity bounds as a function of the encoding size of the system of equations.

Concretely, the papers [31, 33] consider *Markovian binary trees* (MBTs), which are essentially multi-type branching processes where every reproduction rule produces either *exactly* two offspring or zero offspring.[1] Using our terminology, in terms of PPSs, the papers [31, 33], show that if the PPS equations, $x = P(x)$, corresponding to a given Markovian binary tree (MBT) satisfy several additional conditions, namely if they are (i) *positively regular* and (ii) *supercritical*,[2] then Newton's method converges monotonically and "quadratically" to the LFP q^*, starting from the all 0 vector. Specifically, quadratic convergence means that there exists a positive real number $c \in \mathbb{R}_+$, such that if $x^{(k)}$ denotes the k'th Newton iterate then $\|x^{(k+1)} - q^*\| \leq c\|x^{(k)} - q^*\|^2$. However, crucially, the authors of [31, 33] give no upper bounds on the parameter (the "constant") c. Note that c is not an absolute constant, independent of the polynomial system $x = P(x)$, but rather it depends, in some unspecified way, on the size of the PPS (and the given BP or MBT). It is easy to give examples showing that c must depend on the encoding size of the PPS. Thus, as in the "linear convergence" results of Esparza et. al. [16], the "quadratic convergence" results of [31, 33] yield no upper bound at all (not even exponential), as a function of the size of the given branching process (or of the input system of equations $x = P(x)$), on the number of iterations needed to obtain any

[1] This, as we will see, is not a very major restriction. Indeed, as shown in [24], and as we describe in the background Section 2, every multi-type branching process can readily be "reduced" in P-time (for purposes pertaining to extinction probabilities) to one in which every rule generates *at most* two offspring. Equivalently, the corresponding PPSs can be assumed wlog to involve at most quadratic polynomials.

[2] These mean respectively, that (i) the associated moment matrix $M = P'(1)$ is *primitive* (or equivalently that the underlying dependency graph is irreducible and aperiodic), and (ii) that the spectral radius $\rho(M)$ is strictly greater than 1 (which is in fact equivalent, in the positively regular setting, to saying that the LFP q^* is strictly less than 1 in all coordinates).

nontrivial approximation of the extinction probability, e.g., to approximate it within any additive constant error less than 1/2. Let us mention that in this paper we will in fact establish, after suitable qualitative P-time preprocessing of the PPS (to remove those variables x_i with $q_i^* = 1$ or $q_i^* = 0$), a quadratic convergence result for Newton's method on general PPSs with a *very* explicit parameter $c = 2^{14|P|+1}$, which depends single-exponentially on the encoding size $|P|$ of the PPS, $x = P(x)$ (see Lemma 6.1). Moreover, this c is essentially the best possible (up to constant factors in the exponent).

The analysis of Newton's method in the works [31, 33], and in all the prior works on which they build, e.g., [8, 9], all ultimately rest on classic results by Ortega and Rheinbolt [47] (which in turn build on Kantorovich's original work), which show that for certain operators $F(x)$ that satisfy suitable differentiability, monotonicity, and additional conditions, Newton's method converges globally, monotonically and quadratically to a solutions x^* such that $F(x^*) = 0$. However, again, the quadratic convergence results of [47], involve non-singularity assumptions that do not hold in general, and more importantly they involve non-constructive "constants", which do not yield any bounds as a function of the encoding size of the input function $F(x)$.

Other work in the matrix-analytic methods literature ([6, 30]) consider other iterative algorithms for extinction probabilities (and other quantities) for Markovian (binary) trees, but again do not prove any bounds on the number of iterations required for convergence as a function of the input encoding size. Another work by Hautphenne [29], considers decomposable multitype branching processes and gives some criteria for deciding, under certain conditions, whether $q_i^* < 1$. In fact, an earlier result in [24] provides a general P-time algorithm for arbitrary multitype branching processes, and for arbitrary PPSs, for deciding whether $q_i^* < 1$.

An equivalent way to formulate the problem of computing the LFP, q^*, of a PPS, $x = P(x)$, is as a mathematical optimization problem: *minimize:* $\sum_{i=1}^{n} x_i$; *subject to:* $\{P(x) - x \leq 0;\ x \geq 0\}$. This program has a unique optimal solution, which is the LFP q^*. If the resulting constraints were convex, the solution could be computed approximately using convex optimization methods. In general, the PPS constraints are not convex (e.g., $x_2x_3 - x_1 \leq 0$ is not a convex constraint), however for certain restricted subclasses of PPSs they are. This is so for *backbutton processes* which were introduced and studied by Fagin et. al. in [26] and used there to analyze random walks on the web. Backbutton processes constitute a restricted subclass of SCFGs (see [24]). Fagin et. al. applied semidefinite programming to approximate the corresponding termination probabilities for backbutton processes, and used this as a basis for approximating other important quantities associated with them.

Our Results. In this paper we provide the first polynomial time algorithm for computing, to any desired accuracy, the least fixed point solution, q^*, of probabilistic polynomial systems, and thus also provide the first P-time approximation algorithm for extinction probabilities of BPs, and termination probabilities of SCFGs and 1-exit RMCs. The algorithm proceeds roughly as follows:

1. We begin with a preprocessing step, in which we determine all variables x_i which have value 0 or 1 in the LFP q^* and remove them from the system.

2. On the remaining system of equations, $x = P(x)$, with an LFP q^* such that $\mathbf{0} < q^* < \mathbf{1}$, we apply Newton's method, starting at initial vector $x^{(0)} := \mathbf{0}$. Our key result is to show that, once variables x_i with $q_i^* \in \{0, 1\}$ have been removed, Newton's method only requires polynomially many iterations (in fact, only *linearly* many iterations) as a function of both the encoding size of the equation system and of $\log(1/\epsilon)$ to converge to within additive error $\epsilon > 0$ of the vector q^*. To do this, we build on the previous works [24, 16, 25], and extend them with new techniques.

3. The result in the previous step applies to the unit-cost arithmetic RAM model of computation, where we assume that each iteration of Newton's method is carried out in *exact* arithmetic. The problem with this, of course, is that in general after only a linear number of iterations, the number of bits required to represent the rational numbers in Newton's method can be exponential in the input's encoding size. We resolve this by showing, via a careful round-off analysis, that if after each iteration of Newton's method the positive rational numbers in question are all *rounded down* to a suitably long but polynomial encoding length (as a function of both the input size and of the desired error $\epsilon > 0$), then the resulting "approximate" Newton iterations will still be well-defined and will still converge to q^*, within the desired error $\epsilon > 0$, in polynomially (in fact *linearly*) many iterations. The correctness of the rounding relies crucially on the properties of PPSs shown in step 2, and it does not work in general for other types of equation systems.[3]

4. We also subsequently establish *quadratic convergence* results with *explicit*, tame, constants, for the convergence of Newton's method (without rounding) to the LFP of a PPS starting at the $\mathbf{0}$ vector. In particular, we show that linearly many iterations of Newton's method as a function of both $\log(\log(1/\epsilon))$ and the encoding size $|P|$ of the PPS, $x = P(x)$, suffice to converge within additive error $\epsilon > 0$ of the LFP vector q^*, when $\mathbf{0} < q^* < \mathbf{1}$. We use these results to show that the quantitative *decision* problem associated with the LFP of PPSs, namely deciding whether $q_i^* > p$ (or whether $q_i^* < p$) for a given probability p, and given PPS, is decidable in *polynomial time in the unit-cost arithmetic RAM model of computation*, i.e. it is in unit-cost-P. Given the previous hardness result from [24], this implies that the decision problem is complete for unit-cost-P, hence it can be solved in polynomial time in the standard model if and only if unit-cost-P =P. A key role in the quadratic convergence bounds is played by norm bounds that we establish on the inverses of certain crucial matrices that arise in the study of PPSs.

The paper is organized as follows: Section 2 gives basic definitions and background. Section 3 addresses the computation of the LFP of PPSs using Newton's method, and provides a linear upper bound on the number of Newton iterations required. Section 4 uses a suitable rounded version of Newton's method and uses it to provide a polynomial time algorithm for approximating the LFP of a PPS to desired precision in the *Turing model of computation*. Section 5 establishes bounds on the norm of the inverse of a key matrix associated with a PPS. This bound is then used in Section 6 to establish a quadratic convergence result (with fully explicit constants) for Newton's method applied to PPSs, and in turn this quadratic convergence result is also used in Section 6 to show that the quantitative decision problem for the LFP of PPSs can be solved in polynomial time in the *unit-cost arithmetic RAM model of computation*, and thus it is complete for unit-cost-P. We conclude in Section 7, where we also mention some subsequent work done more recently, building on this work.

2 Definitions and Background

A (finite) ***multi-type Branching Process*** (**BP**), $G = (V, R)$, consists of a (finite) set $V = \{S_1, \ldots, S_n\}$ of *types*, and a (finite) set $R = \cup_{i=1}^{n} R_i$ of *rules*, which are partitioned into distinct rule sets, R_i, associated with each type S_i. Each rule $r \in R_i$ has the form $S_i \xrightarrow{p_r} \alpha_r$, where $p_r \in (0, 1]$, and α_r is a finite multiset (possibly the empty multiset) whose elements are in V. Furthermore, for every type S_i, we have $\sum_{r \in R_i} p_r = 1$. The rule $S_i \xrightarrow{p_r} \alpha_r$ specifies the probability with which an

[3] In particular, there are examples of PPSs which *do* have $q_i^* = 1$ for some i, such that this rounding method fails completely because of very severe *ill-conditioning* (see [16]).

entity (or object) of type S_i generates the multiset α_r of offsprings in the next generation. As usual, rule probabilities p_r are assumed to be rational for computational purposes, and are given by their numerator and denominator encoded in binary. Multisets α_r over V can be encoded by giving a vector $v(\alpha_r) \in \mathbb{N}^n$, with the i'th coordinate $v(\alpha_r)_i$ representing the number of elements of type S_i in the multiset α_r. We assume instead that the multisets α_r are represented even more succinctly in *sparse representation*, by specifying only the non-zero coordinates of the vector $v(\alpha_r)$, encoded in binary. The encoding size of the BP is the number of bits needed to specify it as described above.

A BP, $G = (V, R)$, defines a discrete-time stochastic (Markov) process, whose states are multisets over V, or equivalently elements of $\mathbb{N}^n$. If the state at time t is α^t, then the next state α^{t+1} at time $t+1$ is determined by *independently* choosing, for each object of each type S_i in the multiset α^t, a random rule $r \in R_i$ of the form $S_i \stackrel{p_r}{\to} \alpha_r$, according to the probability p_r of that rule, yielding the multiset α_r as the "offspring" of that object in one generation. The multiset α^{t+1} is then given by the *multiset union* of all such offspring multisets, randomly and independently chosen for each object in the multiset α^t. A trajectory (*sample path*) of this stochastic process, starting at time 0 in initial multiset α^0, is a sequence $\alpha^0, \alpha^1, \alpha^2, \ldots$ of multisets over V. Note that if ever the process reaches *extinction*, i.e., if ever $\alpha^t = \{\}$ at some time $t \geq 0$, then $\alpha^{t'} = \{\}$ for all times $t' \geq t$.

Very fundamental quantities associated with a BP, which are a key to many analyses of BPs, are its vector of ***extinction probabilities***, $q^* \in [0,1]^n$, where q_i^* is defined as the probability that, starting with initial multiset $\alpha^0 := \{S_i\}$ at time 0, i.e., starting with a single object of type S_i, the stochastic process eventually reaches extinction, i.e., that $\alpha^t = \{\}$ at some time $t > 0$. From the vector q^*, one can easily compute the extinction probability of the process for any initial population $\mu = \alpha^0$: the extinction probability is simply $\Pi_{i=1}^n (q_i^*)^{\mu_i}$.

Given a BP, $G = (V, R)$, there is a system of polynomial equations in $n = |V|$ variables, $x = P(x)$, that we can associate with G, such that the *least* non-negative solution vector for $x = P(x)$ is the vector of extinction probabilities q^* (see, e.g., [35, 24]). Let us define these equations. For an n-vector of variables $x = (x_1, \ldots, x_n)$, and a vector $v \in \mathbb{N}^n$, we use the shorthand x^v to denote the monomial $x_1^{v_1} \ldots x_n^{v_n}$. Given BP $G = (V, R)$, we define equation $x_i = P_i(x)$ by: $x_i = \sum_{r \in R_i} p_r x^{v(\alpha_r)}$. This yields n polynomial equations in n variables, which we denote by $x = P(x)$. It is not hard to establish that $q^* = P(q^*)$. In fact, q^* is the *least* non-negative solution of $x = P(x)$. In other words, if $q' = P(q')$ for $q' \in \mathbb{R}^n_{\geq 0}$, then $q' \geq q^* \geq 0$, i.e., $q'_i \geq q_i^*$ for all $i = 1, \ldots, n$.

Note that this system of polynomial equations $x = P(x)$ has very special properties. Namely, (I): the coefficients and constant of each polynomial $P_i(x) = \sum_{r \in R_i} p_r x^{v(\alpha_r)}$ are nonnegative, i.e., $p_r \geq 0$ for all r. Furthermore, (II): the coefficients sum to 1, i.e., $\sum_{r \in R_i} p_r = 1$. We call $x = P(x)$ a ***probabilistic polynomial system*** of equations (**PPS**) if it has properties (I) and (II) except that for convenience we weaken (II) and also allow (II′): $\sum_{r \in R_i} p_r \leq 1$. If a system of equations $x = P(x)$ only satisfies (I), then we call it a ***monotone polynomial system*** of equations (**MPS**).

For any PPS, $x = P(x)$, $P(x)$ defines a *monotone* operator $P : [0,1]^n \to [0,1]^n$, i.e., if $y \geq x \geq \mathbf{0}$ then $P(y) \geq P(x)$. For any BP with corresponding PPS $x = P(x)$, q^* is precisely the *least fixed point* (**LFP**) of the monotone operator $P : [0,1]^n \to [0,1]^n$ (see [24]). A MPS, $x = P(x)$, also defines a monotone operator $P : \mathbb{R}^n_{\geq 0} \to \mathbb{R}^n_{\geq 0}$ on the non-negative orthant $\mathbb{R}^n_{\geq 0}$. An MPS need not in general have any solution in $\mathbb{R}^n_{\geq 0}$, but when it does so, it has a *least fixed point* solution $q^* = P(q^*)$ such that $0 \leq q' = P(q')$ implies $q^* \leq q'$.

Note that *any* PPS (with rational coefficients) can be obtained as the system of equations

$x = P(x)$ for a corresponding BP G (with rational rule probabilities), and vice versa.[4] Thus, the computational problem of computing the extinction probabilities of a given BP is equivalent to the problem of computing the least fixed point (LFP) solution q^* of a given PPS, $x = P(x)$. For a PPS $x = P(x)$, we shall use $|P|$ to denote the sum of the number n of variables and the numbers of bits of all the nonzero coefficients and nonzero exponents of all the polynomials in the PPS. (As usual, all rationals are given as the ratios of two integers, and all integers are encoded in binary.) Note that the encoding length of a PPS in sparse representation is at least $|P|$ (and at most $O(|P| \log n)$).

The probabilities q^* can in general be irrational, and even deciding whether $q_i^* \geq 1/2$ is as hard as long standing open problems, including the *square-root sum* problem, which are not even known to be in NP (see [24]). We instead want to approximate q^* within a desired additive error $\epsilon > 0$. In other words, we want to compute a rational vector $v' \in \mathbb{Q}^n \cap [0,1]^n$ such that $\|q^* - v'\|_\infty < \epsilon$.

There are several other related stochastic models, for which the computation of basic probabilities can be similarly expressed as the problem of computing the least fixed point of a PPS. We define two such models here, SCFGs and 1-RMCs. A *Stochastic Context-Free Grammar* (SCFG) is a tuple $G = (T, V, R, S_1)$, where T is a finite set of *terminal* symbols, $V = \{S_1, \ldots, S_k\}$ is a set of *nonterminals*, and $R = \cup_{i=1}^n R_i$ is a set of rules. Each rule $r \in R_i$ has the form $S_i \stackrel{p_r}{\rightarrow} \alpha_r$, where $p_r \in (0,1]$ is the probability of the rule, $\alpha_r \in (V \cup T)^*$ is a string of terminals and nonterminals, and $\sum_{r \in R_i} p_r = 1$ for all i. S_1 is specified as the starting nonterminal. A SCFG G generates a language $L(G) \subseteq T^*$ of terminal strings and associates a probability $p(\tau)$ to every terminal string τ in the language, according to the following stochastic process. Start with the starting nonterminal S_1, pick a rule in R_1 at random (according to the probabilities of the rules) and replace S_1 with the string on the right-hand side of the rule. In general, in each step we have a string $\sigma \in (V \cup T)^*$; take the leftmost nonterminal S_i in the string σ (if there is any), pick a random rule in R_i (according to the probabilities of the rules) and replace this occurrence of S_i in σ by the right-hand side of the rule to obtain a new string σ'. The process stops only when (and if) the current string σ has only terminals. The probability $p(\tau)$ of a terminal string is the probability that the process terminates with the string τ. (The choice of the leftmost nonterminal to replace in each step in the above description is arbitrary and does not affect the probabilities of terminal strings. Any other choice, e.g., rightmost, simultaneous, etc., yields the same probabilities.) The probability of the language $L(G)$ of the SCFG G is $p(L(G)) = \sum_{\tau \in L(G)} p(\tau)$. Note that $L(G)$ is the probability that the stochastic process that we described above, starting with S_1 terminates. More generally, we can define for each nonterminal $S_j \in V$ an associated probability $p(S_j)$, which is the probability that the process starting with S_j terminates. The vector of termination probabilities $p^* = \langle p(S_j) | j = 1, \ldots, n \rangle$ is called the partition function of the SCFG G. This vector is useful in computing other basic quantities of a SCFG (including the probabilities of strings). As with branching processes, we can similarly construct from a SCFG G a PPS $x = P(x)$ that has one variable x_i and one equation for each nonterminal S_i of G, such that the LFP of the PPS is precisely the partition function p^*.

A *1-exit Recursive Markov Chain* (1-RMC) consists of a finite set of components $A_1, \ldots A_k$ where each A_i is a essentially a finite state Markov chain where some of its states represent recursive calls. Formally, each component A_i has a finite set N_i of nodes and a finite set B_i of "boxes" (or supernodes), where each box is labeled by (mapped to) some component. One node en_i is specified

[4] "*non-proper*" PPSs where $\sum_{r \in R_i} p_r < 1$ can be translated easily to BPs by adding an extra dummy type S_{n+1}, with rule $S_{n+1} \stackrel{1}{\rightarrow} \{S_{n+1}, S_{n+1}\}$, so $q_{n+1}^* = 0$, and adding to R_i, for each "non-proper" i, the rule $S_i \stackrel{p_i'}{\rightarrow} \{S_{n+1}, S_{n+1}\}$ with probability $p_i' := (1 - \sum_{r \in R_i} p_r)$. The probabilities q^* for the BP (ignoring $q_{n+1}^* = 0$) give the LFP of the PPS .

as the entry of component A_i and one node ex_i as the exit of A_i.[5] The exit node has no outgoing edges. All other nodes and boxes have outgoing edges with associated probabilities summing to 1. Execution of a 1-RMC starts at some node, e.g. at the entry en_1 of component A_1. When the execution is at a non-exit node v (of some component) then an edge out of v is chosen randomly according to the edge probabilities and the trajectory moves to the tail of the edge. If the execution is at a box b of A_i and b is labeled by component A_j, then the current component A_i is suspended at b and a recursive call to A_j is initiated at its entry node en_j; if and when the call to A_j terminates, i.e., reaches its exit ex_j, the execution of component A_i resumes from box b and follows a random outgoing edge according to the probability distribution of the edges out of b. Note that a call to a component can make further recursive calls, thus at any point in time there is in general a stack (a sequence) of suspended recursive calls, and there can be an arbitrary number of such suspended calls; thus a 1-RMC induces in general an infinite-state Markov chain. The process terminates when the execution reaches the exit of the component of the initial node and there are no suspended recursive calls.

A basic quantity of interest is the termination probability of the 1-RMC for any initial node. We can transform this problem to the problem of computing the LFP of a PPS. Given a 1-RMC, we associate one variable x_u with every node u of every component, and two variables x_b, x'_b for each box b. The exit nodes have value $x_{ex_i} = 1$. The equation for each (non-exit) node u whose outgoing edges (u, v) have probability p_{uv} is $x_v = \sum_v p_{uv} x_v$; the equation for the variable x'_b of a box b is $x'_b = \sum_v p_{bv} x_v$; the equation for the variable x_b of a box b labeled by component A_j with entry en_j is $x_b = x_{en_j} x'_b$. It can be shown then that if the LFP of this PPS is q^*, then the probability that the 1-RMC starting at a node v terminates is precisely q^*_v [24]. Note that the PPS for a 1-RMC has a particularly simple form. All PPS can be put in this form.

A PPS, $x = P(x)$, is said to be in *Simple Normal Form* (SNF) if for every $i = 1, \ldots, n$, the polynomial $P_i(x)$ has one of two forms: (1) Form$_*$: $P_i(x) \equiv x_j x_k$ is simply a quadratic monomial; or (2) Form$_+$: $P_i(x)$ is a *linear* expression $\sum_{j \in \mathcal{C}_i} p_{i,j} x_j + p_{i,0}$, for some rational non-negative coefficients $p_{i,j}$ and $p_{i,0}$, and some index set $\mathcal{C}_i \subseteq \{1, \ldots, n\}$, where $\sum_{j \in \mathcal{C}_i \cup \{0\}} p_{i,j} \leq 1$. We call such a linear equation *non-proper* if $\sum_{j \in \mathcal{C}_i \cup \{0\}} p_{i,j} < 1$. An MPS is said to be in SNF if the same conditions hold except we do not require $\sum_{j \in \mathcal{C}_i \cup \{0\}} p_{i,j} \leq 1$.

Proposition 2.1 (cf. Proposition 7.3 [24])**.** *Every PPS (MPS), $x = P(x)$, can be transformed in P-time to an "*equivalent*" PPS (MPS, respectively), $y = \hat{P}(y)$ in SNF form, such that $|\hat{P}| \in O(|P|)$. More precisely, the variables x are a subset of the variables y, and $y = \hat{P}(y)$ has LFP $\hat{q}^* \in \mathbb{R}^m_{\geq 0}$ iff $x = P(x)$ has LFP $q^* \in \mathbb{R}^n_{\geq 0}$, and projecting $\hat{q}^*$ onto the x variables yields q^*.*

Proof. We prove we can convert any PPS (MPS), $x = P(x)$, to SNF form by adding new auxiliary variables, obtaining a different system of polynomial equations $y = \hat{P}(y)$ with $|\hat{P}|$ linear in $|P|$.

To do this, we simply observe that we can use repeated squaring and Horner's rule to express any monomial x^α via a circuit (straight-line program) with gates $*$, and with the variables x_i as input. Such a circuit will have size $O(m)$ where m is the sum of the numbers of bits of the positive elements in the vector α of exponents. We can then convert such a circuit to a system of equations, by simply replacing the original monomial x^α by a new variable y, and by simply using auxiliary variables in place of the gates of the circuit to "compute" the monomial x^α that the variable y should be equal to.

[5]The restriction to having only one entry, made here for simplicity, is not important; any multi-entry RMC can be efficiently transformed to an 1-entry RMC. The restriction to 1 exit is very important; multi-exit RMCs are more powerful and harder to analyze.

Note that by doing this, every monomial on the RHS of any of the original equations $x_i = P_i(x)$ will have been replaced by a single variable, and thus those original equations will now become Form$_+$ linear equations, and note that all internal gates of the circuit for representing x^α, represented by a variable y_i, give simply the product of two other variables, and thus their corresponding equations are simply of the form $y_i = y_j y_k$, which constitutes a Form$_*$ equation.

Importantly, note that the system of equations so obtained will still remain a system of monotone (and respectively, probabilistic) polynomial equations, if the original system was monotone (respectively, probabilistic), because each new auxiliary variable y_i, that we introduce (which acts as a gate in the circuit for the monomial x^α), will be associated with an equation of the form $y_i = y_j y_k$, which indeed is both a monotone and probabilistic equation.

Furthermore, the new system of equations $y = \hat{P}(y)$ has the property that (a) any solution $\hat{q} \in \mathbb{R}^n_{\geq 0}$ of $y = \hat{P}(y)$, when projected on to the x variables, yields a solution $q \in \mathbb{R}^n_{\geq 0}$ to the original system of equations, $x = P(x)$, and (b) any solution $q \in \mathbb{R}^n_{\geq 0}$ to the original system of equations $x = P(x)$ yields a *unique* solution $\hat{q}$ to the expanded system of equations, $y = \hat{P}(y)$, by uniquely solving for the values of the new auxiliary variables using their equations (which are derived from the arithmetic circuit).

The $O(|P|)$ bound that is claimed for $|\hat{P}|$ follows easily from the fact that the circuit representing each monomial x^α has size $O(m)$, where m is the sum of the numbers of bits of the positive elements in the vector α. □

Proposition 2.2 ([24]). *There is a P-time algorithm that, given a PPS, $x = P(x)$, over n variables, with LFP $q^* \in \mathbb{R}^n_{\geq 0}$, determines for every $i = 1, \ldots, n$ whether $q^*_i = 0$ or $q^*_i = 1$ or $0 < q^*_i < 1$.*

Thus, for every PPS, we can detect in P-time all the variables x_j such that $q^*_j = 0$ or $q^*_j = 1$. We can then remove these variables and their corresponding equation $x_j = P_j(x)$, and substitute their values on the right hand sides (RHS) of the remaining equations. This yields a new PPS, $x' = P'(x')$, where its LFP solution, q'^* (which corresponds to the remaining coordinates of q^*) satisfies $\mathbf{0} < q'^* < \mathbf{1}$, where $\mathbf{0}, \mathbf{1}$ denote the all-0 and all-1 vectors respectively.

We can thus henceforth assume, w.l.o.g., that any given PPS, $x = P(x)$, is in SNF form and has an LFP solution q^* such that $\mathbf{0} < q^* < \mathbf{1}$.

For a MPS or PPS, $x = P(x)$, its variable *dependency graph* is defined to be the digraph $H = (V, E)$, with vertices $V = \{x_1, \ldots, x_n\}$, such that $(x_i, x_j) \in E$ iff in $P_i(x) = \sum_{r \in R_i} p_r x^{v(\alpha_r)}$ there is a coefficient $p_r > 0$ such that $v(\alpha_r)_j > 0$. Intuitively, $(x_i, x_j) \in E$ means that x_i "depends directly" on x_j. We say that x_i ***depends on*** x_j (directly or indirectly) if there is a path in the dependency graph starting at x_i and ending at x_j.

A MPS or PPS, $x = P(x)$, is called ***strongly connected*** if its dependency graph H is strongly connected, that is, every node has a path to every other node. As in [24], for analysing PPSs we will find it very useful to decompose the PPS based on the *strongly connected components* (SCCs) of its variable dependency graph.

3 Polynomial upper bounds for Newton's method on PPSs

To find a solution for a differentiable system of equations $F(x) = \mathbf{0}$, in n variables, *Newton's method* uses the following iteration scheme: start with some initial vector $x^{(0)} \in \mathbb{R}^n$, and for $k > 0$ let: $x^{(k+1)} := x^{(k)} - F'(x^{(k)})^{-1}(F(x^{(k)}))$, where $F'(x)$ is the Jacobian matrix of $F(x)$.

Let $x = P(x)$ be a given PPS (or MPS) in n variables. Let $B(x) := P'(x)$ denote the Jacobian matrix of $P(x)$. In other words, $B(x)$ is an $n \times n$ matrix such that $B(x)_{i,j} = \frac{\partial P_i(x)}{\partial x_j}$. Using Newton iteration, starting at n-vector $x^{(0)} := \mathbf{0}$, yields the following iteration:

$$x^{(k+1)} := x^{(k)} + (I - B(x^{(k)}))^{-1}(P(x^{(k)}) - x^{(k)})) \tag{1}$$

For a vector $z \in \mathbb{R}^n$, assuming that matrix $(I - B(z))$ is non-singular, we define a single iteration of Newton's method for $x = P(x)$ on z via the following operator:

$$\mathcal{N}_P(z) := z + (I - B(z))^{-1}(P(z) - z) \tag{2}$$

It was shown in [24] that for any MPS, $x = P(x)$, with LFP $q^* \in \mathbb{R}^N_{\geq 0}$, if we first find and remove the variables that have value 0 in the LFP, q^*, and apply a decomposed variant of Newton's method that decomposes the system according to the strongly connected components (SCCs) of the dependency graph and processes them bottom-up, then the values converge *monotonically* to q^*. PPSs are a special case of MPSs, so the same applies to PPSs. In [16], it was pointed out that if $q^* > 0$, i.e., after we remove the variables x_i where $q^*_i = 0$, decomposition into SCCs isn't strictly necessary. (Decomposition is nevertheless very useful in practice, as well as in the theoretical analysis, including in this paper.) Thus:

Proposition 3.1 (cf. Theorem 6.1 of [24] and Theorem 4.1 of [16])**.** *Let $x = P(x)$ be a MPS, with LFP $q^* > \mathbf{0}$. Then starting at $x^{(0)} := \mathbf{0}$, the Newton iterations $x^{(k+1)} := \mathcal{N}_P(x^{(k)})$ are well defined and monotonically converge to q^*, i.e. $\lim_{k\to\infty} x^{(k)} = q^*$, and $x^{(k+1)} \geq x^{(k)} \geq \mathbf{0}$ for all $k \geq 0$.*

We will actually establish an extension of this result in this paper, because in Section 4 we will need to show that even when each iterate is suitably *rounded off*, the rounded Newton iterations are all well-defined and converge to q^*. The main goal of this section is to show that for PPSs, $x = P(x)$, with LFP $0 < q^* < 1$, polynomially many iterations of Newton's method, *using exact rational arithmetic*, suffice, as a function of $|P|$ and j, to compute q^* to within additive error $1/2^j$. In fact, we show a much stronger *linear* upper bound with small explicit constants:

Theorem 3.2 (**Main Theorem of Section 3**)**.** *Let $x = P(x)$ be any PPS in SNF form, with LFP q^*, such that $\mathbf{0} < q^* < \mathbf{1}$. If we start Newton iteration at $x^{(0)} := \mathbf{0}$, with $x^{(k+1)} := \mathcal{N}_P(x^{(k)})$, then for any integer $j \geq 0$ the following inequality holds: $\|q^* - x^{(j+4|P|)}\|_\infty \leq 2^{-j}$.*

Before we embark on the proof of the theorem, we summarize the two key properties that lead to the theorem. The first property concerns the improvement in the gap $q^* - x^{(k)}$ between the LFP q^* and the Newton iterate $x^{(k)}$ from one iteration to the next, as compared to the gap $\mathbf{1} - q^*$ between the all-1 vector $\mathbf{1}$ and the LFP q^*. Specifically, we show that if $q^* - x^{(k)} \leq \lambda(\mathbf{1} - q^*)$ for some $\lambda > 0$ then $q^* - x^{(k+1)} \leq (\lambda/2)(\mathbf{1} - q^*)$. Or putting it another way, the maximum ratio between corresponding coordinates of $q^* - x^{(k)}$ and $\mathbf{1} - q^*$, i.e., $\max_i \frac{q^*_i - x^{(k)}_i}{1 - q^*_i}$, shrinks by a factor of 2 (at least) in each iteration. Note that it is crucial here that $q^* < \mathbf{1}$, hence $1 - q^*_i > 0$ for all i. The second key property is that if $q^* < \mathbf{1}$ then the gap $\mathbf{1} - q^*$ is not too small in any coordinate, specifically $\min_i(1 - q^*_i) \geq 2^{-4|P|}$.

These two key properties imply the theorem. Initially, for $k = 0$, $x^{(0)} = \mathbf{0}$, and the ratio $\max_i \frac{q^*_i - x^k_i}{1 - q^*_i}$ is less than $1/\min_i(1 - q^*_i) \leq 2^{4|P|}$. The ratio shrinks by a factor of 2 in each iteration, so after $j + 4|P|$ iterations, it is at most 2^{-j}, hence $\max_i(q^*_i - x^{j+4|P|}_i) \leq 2^{-j}$.

We will prove the two properties in turn through a sequence of lemmas. The first lemma states a basic fact, which we will use often, that exploits the assumption that the system $x = P(x)$ is in SNF form, thus P is quadratic and its matrix of derivatives (the Jacobian B) is linear.

Lemma 3.3. *Let $x = P(x)$ be a MPS, with n variables, in SNF form, and let $a, b \in \mathbb{R}^n$. Then:*

$$P(a) - P(b) = B(\frac{a+b}{2})(a-b) = \frac{B(a)+B(b)}{2}(a-b)$$

Proof. Let the function $f : \mathbb{R} \to \mathbb{R}^n$ be given by $f(t) := ta + (1-t)b = b + t(a-b)$. Define $G(t) := P(f(t))$.

From the fundamental theorem of calculus, and using the matrix form of the chain rule from multi-variable calculus (see, e.g., [3] Section 12.10), we have:

$$P(a) - P(b) = G(1) - G(0) = \int_0^1 B(f(t))(a-b)\,dt$$

By linearity, we can just take out $(a-b)$ from the integral as a constant, and we get:

$$P(a) - P(b) = (\int_0^1 B(ta + (1-t)b)\,dt)(a-b)$$

We need to show that

$$\int_0^1 B(ta + (1-t)b)\,dt = B(\frac{a+b}{2}) = \frac{B(a)+B(b)}{2}$$

Since all monomials in $P(x)$ have degree at most 2, each entry of the Jacobian matrix $B(x)$ is a polynomial of degree 1 over variables in x. For any integers i, j, with $1 \le i \le n$, $1 \le j \le n$, there are thus real values α_{ij} and β_{ij} with

$$(B(ta + (1-t)b))_{ij} = \alpha_{ij} + \beta_{ij} t$$

Then

$$(\int_0^1 B(ta + (1-t)b)\,dt)_{ij} = \int_0^1 (\alpha_{ij} + \beta_{ij} t)\,dt = \alpha_{ij} + \frac{\beta_{ij}}{2}$$

$$(B(\frac{a+b}{2}))_{ij} = \alpha_{ij} + \frac{\beta_{ij}}{2}$$

$$\left(\frac{B(a)+B(b)}{2}\right)_{ij} = \frac{1}{2}((\alpha_{ij} + \beta_{ij}) + \alpha_{ij}) = \alpha_{ij} + \frac{\beta_{ij}}{2}.$$

□

We apply this to relate the gap $q^* - z$ in one iteration to the gap $q^* - \mathcal{N}_P(z)$ in the next iteration:

Lemma 3.4. *Let $x = P(x)$ be a MPS in SNF form. Let $z \in \mathbb{R}^n$ be any vector such that $(I - B(z))$ is non-singular, and thus $\mathcal{N}_P(z)$ is defined. Then:*

$$q^* - \mathcal{N}_P(z) = (I - B(z))^{-1} \frac{B(q^*) - B(z)}{2}(q^* - z)$$

Proof. Lemma 3.3, applied to q^* and z, gives: $q^* - P(z) = \frac{B(q^*)+B(z)}{2}(q^* - z)$.
Rearranging, we get:

$$P(z) - z = (I - \frac{B(q^*) + B(z)}{2})(q^* - z) \tag{3}$$

Replacing $(P(z) - z)$ in equation (2) by the right hand side of equation (3) and subtracting both sides of (2) from q^*, gives:

$$\begin{aligned} q^* - \mathcal{N}_P(z) &= (q^* - z) - (I - B(z))^{-1}(I - \frac{B(q^*) + B(z)}{2})(q^* - z) \\ &= (I - B(z))^{-1}(I - B(z))(q^* - z) - (I - B(z))^{-1}(I - \frac{B(q^*) + B(z)}{2})(q^* - z) \\ &= (I - B(z))^{-1}((I - B(z)) - (I - \frac{B(q^*) + B(z)}{2}))(q^* - z) \\ &= (I - B(z))^{-1}(\frac{B(q^*) - B(z)}{2})(q^* - z) \end{aligned}$$

□

To prove their exponential upper bounds for *strongly connected* PPSs, [16] used the notion of a *cone vector* for the matrix $B(q^*)$, that is a vector $d > 0$ such that $B(q^*)d \leq d$. For a strongly connected MPS, $x = P(x)$, with $q^* > 0$, the matrix $B(q^*) \geq 0$ is irreducible, and thus has a positive eigenvector. They used this eigenvector as their cone vector $d > 0$. However, such an eigenvector yields only weak (exponential) bounds. Instead, we show there is a different cone vector for $B(q^*)$, and even for $B(\frac{1}{2}(\mathbf{1} + q^*))$, that works for arbitrary (not necessarily strongly-connected) PPSs; namely, the vector $\mathbf{1} - q^*$ is such a cone vector:

Lemma 3.5. *If $x = P(x)$ is a PPS in n variables, in SNF form, with LFP $\mathbf{0} < q^* < \mathbf{1}$, and where $P(x)$ has Jacobian $B(x)$, then $\forall z \in \mathbb{R}^n$ such that $\mathbf{0} \leq z \leq \frac{1}{2}(\mathbf{1} + q^*)$: $B(z)(\mathbf{1} - q^*) \leq (\mathbf{1} - q^*)$. In particular, $B(\frac{1}{2}(\mathbf{1} + q^*))(\mathbf{1} - q^*) \leq (\mathbf{1} - q^*)$, and $B(q^*)(\mathbf{1} - q^*) \leq (\mathbf{1} - q^*)$.*

Proof. Lemma 3.3 applied to $\mathbf{1}$ and q^* gives: $P(\mathbf{1}) - P(q^*) = P(\mathbf{1}) - q^* = B(\frac{1}{2}(\mathbf{1} + q^*))(\mathbf{1} - q^*)$. But note that $P(\mathbf{1}) \leq \mathbf{1}$, because for any PPS, since the nonnegative coefficients of each polynomial $P_i(x)$ sum to ≤ 1, $P(x)$ maps $[0, 1]^n$ to $[0, 1]^n$. Thus $\mathbf{1} - q^* \geq P(\mathbf{1}) - q^* = B(\frac{1}{2}(\mathbf{1} + q^*))(\mathbf{1} - q^*)$. Now observe that for $0 \leq z \leq \frac{1}{2}(\mathbf{1} + q^*)$, $B(\frac{1}{2}(\mathbf{1} + q^*)) \geq B(z) \geq 0$, because the entries of Jacobian $B(x)$ have nonnegative coefficients. Thus since $(\mathbf{1} - q^*) \geq 0$, we have $(\mathbf{1} - q^*) \geq B(z)(\mathbf{1} - q^*)$. □

For a square matrix A, let $\rho(A)$ denote the spectral radius of A. We need the following basic fact:

Lemma 3.6 (see, e.g., [37])**.** *If A is a square nonnegative matrix with $\rho(A) < 1$ then $(I - A)$ is non-singular, the series $\sum_{k=0}^{\infty} A^k$ converges, $(I - A)^{-1} = \sum_{k=0}^{\infty} A^k$ and is nonnegative.*

Theorem 3.7. *For any PPS, $x = P(x)$, in SNF form, if we have $\mathbf{0} < q^* < \mathbf{1}$, then for all $\mathbf{0} \leq z \leq q^*$, $\rho(B(z)) < 1$ and $(I - B(z))^{-1}$ exists and is nonnegative.*

Proof. For all $\mathbf{0} \leq z \leq q^*$, $B(z)$ is a nonnegative matrix, and since the entries of the Jacobian matrix $B(x)$ have nonnegative coefficients, $B(x)$ is monotone in x, i.e., if $\mathbf{0} \leq z \leq q^*$, then $0 \leq B(z) \leq B(q^*)$, and thus by basic facts about non-negative matrices $\rho(B(z)) \leq \rho(B(q^*))$. Thus by Lemma 3.6 it suffices to establish that $\rho(B(q^*)) < 1$. We will first prove this for *strongly connected* PPSs:

Lemma 3.8. *For any strongly connected PPS, $x = P(x)$, in SNF form with LFP q^*, such that $\mathbf{0} < q^* < \mathbf{1}$, we have $\rho(B(q^*)) < 1$.*

Proof. If the Jacobian $B(x)$ is constant, then $B(q^*) = B(\mathbf{1}) = B$. In this case, B is actually an irreducible substochastic matrix, and since we have removed all variables x_i such that $q_i^* = 0$, it is easy to see that some polynomial $P_i(x)$ must have contained a positive constant term, and therefore, in the (constant) Jacobian matrix B there is some row whose entries sum to < 1. Since B is also irreducible, we then clearly have that $\lim_{m\to\infty} B^m = 0$. But this is equivalent to saying that $\rho(B) < 1$. Thus we can assume that the Jacobian $B(x)$ is non-constant. By Lemma 3.5:

$$B(\frac{1}{2}(\mathbf{1} + q^*))(\mathbf{1} - q^*) \leq (\mathbf{1} - q^*)$$

We have $\mathbf{1} - q^* > 0$, and $B(\frac{1}{2}(\mathbf{1} + q^*)) \geq 0$. Thus, by induction, for any positive integer power k, we have

$$B(\frac{1}{2}(\mathbf{1} + q^*))^k(\mathbf{1} - q^*) \leq (\mathbf{1} - q^*) \tag{4}$$

Now, since $B(x)$ is non-constant, and $B(x)$ is monotone in x, and since $q^* < \frac{1}{2}(\mathbf{1} + q^*)$, we have $B(q^*) \leq B(\frac{1}{2}(\mathbf{1}+q^*))$ and furthermore there is some entry (i, j) such that $B(q^*)_{i,j} < B(\frac{1}{2}(\mathbf{1}+q^*))_{i,j}$, it follows that:

$$(B(q^*)(\mathbf{1} - q^*))_i < (B(\frac{1}{2}(\mathbf{1} + q^*))(\mathbf{1} - q^*))_i \leq (\mathbf{1} - q^*)_i$$

Therefore, since $B(q^*)$ is irreducible, it follows that for any coordinate r there exists a power $k \leq n$ such that $(B(q^*)^k(\mathbf{1} - q^*))_r < (\mathbf{1} - q^*)_r$. Therefore, $B(q^*)^n(\mathbf{1} - q^*) < (\mathbf{1} - q^*)$. Thus, there exists some $0 < \beta < 1$, such that $B(q^*)^n(\mathbf{1} - q^*) \leq \beta(\mathbf{1} - q^*)$. Thus, by induction on m, for all $m \geq 1$, we have $B(q^*)^{nm}(\mathbf{1} - q^*) \leq \beta^m(\mathbf{1} - q^*)$. But $\lim_{m\to\infty} \beta^m = 0$, and thus since $(\mathbf{1} - q^*) > 0$, it must be the case that $\lim_{m\to\infty} B(q^*)^{nm} = 0$ (in all coordinates). But this last statement is equivalent to saying that $\rho(B(q^*)) < 1$. □

Now we can proceed to arbitrary PPSs. We want to show that $\rho(B(q^*)) < 1$. Consider an eigenvector $v \in \mathbb{R}^n_{\geq 0}$, $v \neq 0$, of $B(q^*)$, associated with the eigenvalue $\rho(B(q^*))$, with $B(q^*)v = \rho(B(q^*))v$. Such an eigenvector exists by standard fact in Perron-Frobenius theory (see, e.g., Theorem 8.3.1 [37]).

Consider any subset $S \subseteq \{1, \ldots, n\}$ of variable indices, and let $x_S = P_S(x_S, x_{D_S})$ denote the subsystem of $x = P(x)$ associated with the vector x_S of variables in set S, where x_{D_S} denotes the variables not in S. Note that $x_S = P_S(x_S, q^*_{D_S})$ is itself a PPS. We call S *strongly connected* if $x_S = P_S(x_S, q^*_{D_S})$ is a strongly connected PPS.

By Lemma 3.8, for any such strongly connected PPS given by indices S, if we define its Jacobian by $B_S(x)$, then $\rho(B_S(q^*)) < 1$. If S defines a bottom strongly connected component that depends on no other components in the system $x = P(x)$, then we would have that $B_S(q^*)v_S = \rho(B(q^*))v_S$ where v_S is the subvector of v with coordinates in S. Unfortunately v_S might in general be the zero vector. However, if we take S to be a strongly connected component that has $v_S \neq 0$ and such that the SCC S only depends on SCCs S' with $v_{S'} = 0$, then we still have $B_S(q^*)v_S = \rho(B(q^*))v_S$. Thus, by another standard fact from Perron-Frobenius theory (see Theorem 8.3.2 of [37]), $\rho(B_S(q^*)) \geq \rho(B(q^*))$. But since $\rho(B_S(q^*)) < 1$, this implies $\rho(B(q^*)) < 1$. □

Note that Theorem 3.7 tells us, in particular, that for *every* z (including q^*), such that $\mathbf{0} \leq z \leq q^*$, the Newton iteration $\mathcal{N}_P(z)$ is well-defined. This will be important in Section 4.

We need the following Lemma from [16]. (To be self-contained, and to clarify our assumptions, we provide a short proof.)

Lemma 3.9 (Lemma 5.4 from [16])**.** *Let $x = P(x)$ be a MPS, in SNF form, with LFP $q^* \geq 0$. Let $B(x)$ denote the Jacobian matrix of $P(x)$. For any positive vector $\boldsymbol{d} \in \mathbb{R}^n_{>0}$ that satisfies $B(q^*)\boldsymbol{d} \leq \boldsymbol{d}$, any positive real value $\lambda > 0$, and any nonnegative vector $z \in \mathbb{R}^n_{\geq 0}$, if $q^* - z \leq \lambda \boldsymbol{d}$, and $(I - B(z))^{-1}$ exists and is nonnegative, then $q^* - \mathcal{N}_P(z) \leq \frac{\lambda}{2}\boldsymbol{d}$.*

Proof. By Lemma 3.4, $q^* - \mathcal{N}_P(z) = (I - B(z))^{-1}\frac{1}{2}(B(q^*) - B(z))(q^* - z)$. Note that matrix $(I - B(z))^{-1}\frac{1}{2}(B(q^*) - B(z))$ is nonnegative: we assumed $(I - B(z))^{-1} \geq 0$ and the positive coefficients in $P(x)$ and in $B(x)$ mean $(B(q^*) - B(z)) \geq 0$. This and the assumption that $q^* - z \leq \lambda \mathbf{d}$ yields: $q^* - \mathcal{N}_P(z) \leq (I - B(z))^{-1}\frac{1}{2}(B(q^*) - B(z))\lambda\mathbf{d}$. We can rearrange as follows:

$$\begin{aligned} q^* - \mathcal{N}_P(z) &\leq (I - B(z))^{-1}\frac{1}{2}(B(q^*) - B(z))\lambda\mathbf{d} \\ &= (I - B(z))^{-1}\frac{1}{2}((I - B(z)) - (I - B(q^*)))\lambda\mathbf{d} \\ &= \frac{\lambda}{2}(I - (I - B(z))^{-1}(I - B(q^*)))\mathbf{d} \\ &= \frac{\lambda}{2}\mathbf{d} - \frac{\lambda}{2}(I - B(z))^{-1}(I - B(q^*))\mathbf{d} \end{aligned}$$

If we can show that $\frac{\lambda}{2}(I - B(z))^{-1}(I - B(q^*))\mathbf{d} \geq 0$, we are done. By assumption: $(I - B(q^*))\mathbf{d} \geq 0$, and since we assumed $(I - B(z))^{-1} \geq 0$ and $\lambda > 0$, we have: $\frac{\lambda}{2}(I - B(z))^{-1}(I - B(q^*))\mathbf{d} \geq 0$. □

Lemmas 3.5 and 3.9 (with Theorem 3.7) imply the first key property we mentioned after the statement of main theorem of this section, Theorem 3.2. Namely, the vector $d = \mathbf{1} - q^*$ is a cone vector for $B(q^*)$, and hence if $q^* - x^{(k)} \leq \lambda(\mathbf{1} - q^*)$ for some $\lambda > 0$ then $q^* - x^{(k+1)} \leq \frac{\lambda}{2}(\mathbf{1} - q^*)$. For a vector $b \in \mathbb{R}^n$, we shall use the following notation: $b_{\min} = \min_i b_i$, and $b_{\max} = \max_i b_i$.

Corollary 3.10. *Let $x = P(x)$ be a PPS, in SNF form, with LFP $0 < q^* < 1$, and let $B(x)$ be the Jacobian matrix for $P(x)$. Suppose there is a vector $d \in \mathbb{R}^n$, $\mathbf{0} < d \leq \mathbf{1}$, such that $B(q^*)d \leq d$. For any positive integer $j > 0$, if we perform Newton's method starting at $x^{(0)} := \mathbf{0}$, then: $\|q^* - x^{(j - \lfloor \log_2 d_{min} \rfloor)}\|_\infty \leq 2^{-j}$.*

Proof. By induction on k, we show $q^* - x^{(k)} \leq 2^{-k}\frac{1}{d_{\min}}d$. For the base case, $k = 0$, since $d > 0$, $\frac{1}{d_{\min}}d \geq \mathbf{1} \geq q^* = q^* - x^{(0)}$. For $k > 0$, apply Lemma 3.9, setting $z := x^{(k-1)}$, $\lambda := \frac{1}{d_{\min}}2^{-(k-1)}$ and $\mathbf{d} := d$. This yields $q^* - x^{(k)} \leq \frac{\lambda}{2}\mathbf{d} = 2^{-k}\frac{1}{d_{\min}}d$. Since we assume $\|d\|_\infty \leq 1$, we have $\|2^{-(j - \lfloor \log_2 d_{\min} \rfloor)}\frac{1}{d_{\min}}d\|_\infty \leq 2^{-j}$, and thus $\|q^* - x^{(j - \lfloor \log_2 d_{\min} \rfloor)}\|_\infty \leq 2^{-j}$. □

Lemma 3.11. *For a PPS in SNF form, with LFP q^*, where $\mathbf{0} < q^* < \mathbf{1}$, if we start Newton iteration at $x^{(0)} := \mathbf{0}$, then:*

$$\|q^* - x^{(j + \lceil (\log_2 \frac{(\mathbf{1} - q^*)_{max}}{(\mathbf{1} - q^*)_{min}}) \rceil)}\|_\infty \leq 2^{-j}.$$

Proof. For $d := \frac{\mathbf{1} - q^*}{\|\mathbf{1} - q^*\|_\infty}$, $d_{min} = \frac{(\mathbf{1} - q^*)_{\min}}{(\mathbf{1} - q^*)_{\max}}$. By Lemma 3.5, $B(q^*)d \leq d$. Apply Corollary 3.10. □

Thus, in order to bound the number of iterations needed to achieve additive error 2^{-j}, it suffices to bound the ratio of the maximum and minimum coordinates of $1 - q^*$. This is easier to do for strongly connected PPS.

Lemma 3.12. *For a strongly connected PPS, $x = P(x)$, with LFP q^*, where $0 < q^* < 1$, for any two coordinates k, l of $\mathbf{1} - q^*$:*

$$\frac{(\mathbf{1} - q^*)_k}{(\mathbf{1} - q^*)_l} \geq 2^{-(2|P|)}.$$

Proof. Lemma 3.5 says that $B(\frac{1}{2}(\mathbf{1}+q^*))(\mathbf{1}-q^*) \leq (\mathbf{1}-q^*)$. Since every entry of the vector $\frac{1}{2}(\mathbf{1}+q^*))$ is $\geq 1/2$, every non-zero entry of the matrix $B(\frac{1}{2}(\mathbf{1}+q^*))$ is at least $1/2$ times a coefficient of some monomial in some polynomial $P_i(x)$ of $P(x)$. Moreover, $B(\frac{1}{2}(\mathbf{1}+q^*))$ is irreducible. Calling the entries of $B(\frac{1}{2}(\mathbf{1}+q^*))$, $b_{i,j}$, we have a sequence of *distinct* indices, $i_1, i_2, \ldots, i_m$, with $l = i_1$, $k = i_m$, $m \leq n$, where each $b_{i_j i_{j+1}} > 0$. (Just take the "shortest positive path" from l to k.) For any j:

$$(B(\frac{1}{2}(\mathbf{1}+q^*))(\mathbf{1}-q^*))_{i_{j+1}} \geq b_{i_j i_{j+1}}(\mathbf{1}-q^*)_j$$

Using Lemma 3.5 again, $(\mathbf{1}-q^*)_{i_{j+1}} \geq b_{i_j i_{j+1}}(\mathbf{1}-q^*)_{i_j}$. By simple induction: $(\mathbf{1}-q^*)_k \geq (\prod_{j=1}^{l-1} b_{i_j i_{j+1}})(\mathbf{1}-q^*)_l$. Note that $|P|$ includes the encoding size of each positive coefficient of every polynomial $P_i(x)$. We argued before that each $b_{i_j i_{j+1}} \geq c_i/2$ for some coefficient $c_i > 0$ of some monomial in $P_i(x)$. Therefore, since each such c_i is a distinct coefficient that is accounted for in $|P|$, we must have $\prod_{j=1}^{l-1} b_{i_j i_{j+1}} \geq 2^{-(|P|+n)} \geq 2^{-(2|P|)}$, and thus we have: $(\mathbf{1}-q^*)_k \geq 2^{-(2|P|)}(\mathbf{1}-q^*)_l$. □

Combining Lemma 3.11 with Lemma 3.12 establishes the following:

Theorem 3.13. *For a strongly connected PPS, $x = P(x)$ in n variables, in SNF form, with LFP q^*, such that $\mathbf{0} < q^* < \mathbf{1}$, if we start Newton iteration at $x^{(0)} := \mathbf{0}$, then: $\|q^* - x^{(j+2|P|)}\|_\infty \leq 2^{-j}$.*

To get a polynomial upper bound on the number of iterations of Newton's method for general PPSs, we can apply Lemma 3.11 combined with a Lemma in [25] (Lemma 7.2 of [25]), which implies that for a PPS $x = P(x)$ with n variables, in SNF form, with LFP q^*, where $q^* < \mathbf{1}$, $(\mathbf{1} - q^*)_{\min} \geq 1/2n2^{|P|^c}$ for some constant c. Instead, we prove the following much stronger result:

Theorem 3.14. *For a PPS, $x = P(x)$ in n variables, in SNF form, with LFP q^*, such that $\mathbf{0} < q^* < \mathbf{1}$, for all $i = 1, \ldots, n$: $\mathbf{1} - q_i^* \geq 2^{-4|P|}$. In other words, $\|q^*\|_\infty \leq 1 - 2^{-4|P|}$.*

We shall prove Theorem 3.14 in subsection 3.1 below. We thus get the Main Theorem of this section:

*Proof of Theorem 3.2 (**Main Theorem of Sec. 3**).* By Lemma 3.11, $\|q^* - x^{(j+\lceil(\log \frac{(\mathbf{1}-q^*)_{\max}}{(\mathbf{1}-q^*)_{\min}})\rceil)}\|_\infty \leq 2^{-j}$. But by Theorem 3.14, $\lceil(\log \frac{(\mathbf{1}-q^*)_{max}}{(\mathbf{1}-q^*)_{min}})\rceil \leq \lceil \log \frac{1}{(\mathbf{1}-q^*)_{min}} \rceil \leq \lceil \log 2^{4|P|} \rceil = 4|P|$. □

In Section 6 we shall extend Theorem 3.2, to show that, given a PPS, $x = P(x)$, with LFP $0 < q^* < 1$, if we start Newton iteration at $x^{(0)} := \mathbf{0}$, then for all $i \geq 1$, $\|q^* - x^{(18|P|+2+i)}\|_\infty \leq \frac{1}{2^{2^i}}$. We then use this (explicit) "quadratic convergence" result to show that the quantitative ***decision problem*** for the LFP q^* of PPSs, which asks, given a PPS $x = P(x)$ over n variables, and given a rational number $r \in [0, 1]$, decide whether $q_i^* > r$, is decidable *in the unit-cost arithmetic RAM model of computation* in polynomial time.

3.1 Proof of Theorem 3.14

As ingredients of our proof, we will need the following two basic linear algebra lemmas:

Lemma 3.15. *Let A be a non-singular $n \times n$ matrix with rational entries. If the product of the denominators of all these entries is m, then*

$$\|A^{-1}\|_\infty \leq nm\|A\|_\infty^n$$

Proof. The i,jth entry of A^{-1} satisfies:

$$(A^{-1})_{ij} = \frac{\det(M_{ij})}{\det(A)}$$

where M_{ij} is the i,jth minor of A, made by deleting row i and column j. $\|M_{ij}\|_\infty \leq \|A\|_\infty$ as we've removed entries from rows. We always have $|\det(M_{ij})| \leq \|M_{ij}\|_\infty^n$ (see, e.g., [37] page 351), so:

$$|(A^{-1})_{ij}| \leq \frac{\|A\|_\infty^n}{|\det(A)|} \ . \tag{5}$$

Meanwhile $\det(A)$ is a non-zero rational number (because by assumption A is non-singular). If we consider the expansion for the determinant $\det(A) = \sum_\sigma \mathrm{sgn}\sigma \prod_{i=1}^n a_{i\sigma(i)}$, then the denominator of each term $\prod_{i=1}^n a_{i\sigma(i)}$ is a product of denominators of distinct entries $a_{i\sigma(i)}$ and therefore divides m. Since every term can thus be rewritten with denominator m, the sum can also be written with denominator m, and therefore $|\det(A)| \geq \frac{1}{m}$. Thus, plugging into inequality (5), we have:

$$|(A^{-1})_{ij}| \leq m\|A\|_\infty^n \ .$$

Taking the maximum row sum $\|A^{-1}\|_\infty$,

$$\|A^{-1}\|_\infty \leq nm\|A\|_\infty^n \ .$$

□

Lemma 3.16. *Suppose we have an equation $Ax = b$, with A a singular $n \times n$ matrix, b a non-zero vector, and we know that $Ax = b$ has a solution. Then it must have a solution of the form $A'^{-1}b$ where A' is a non-singular matrix generated from A by replacing some rows with rows that have a single 1 entry and the rest 0.*

Proof. If A has rank $r < n$, then there are linearly independent vectors $a_1, a_2, \ldots, a_r$ such that $a_1^T, a_2^T, \ldots, a_r^T$ are rows of A and other rows of A are linear combinations of these. Let $e_1, e_2, \ldots, e_n$ be the canonical basis of $\mathbb{R}^n$, i.e. each e_i has ith coordinate 1 and the rest 0. By the well known fact that the set of linearly independent subsets of a vector space form a matroid, and in particular satisfy the exchange property of a matroid (see any good linear algebra or combinatorics text, e.g,. [12], Proposition 12.8.2) , we know there is a basis for $\mathbb{R}^n$ of the form $\{a_1, a_2, \ldots, a_r, e_{i_{r+1}}, e_{i_{r+2}}, \ldots, e_{i_n}\}$ for some choice of $i_{r+1}, i_{r+2}, \ldots, i_n$. We form a matrix A' with elements of this basis as rows by starting with A and keeping r rows corresponding to $a_1^T, a_2^T, \ldots a_r^T$, and replacing the others in some order with $e_{i_{r+1}}^T, e_{i_{r+2}}^T, \ldots, e_{i_n}^T$. Specifically, there is a permutation σ of $\{1, \ldots, n\}$ such that if $1 \leq k \leq r$, the $\sigma(k)$'th row of A' and A are a_k^T and if $r < k \leq n$, the $\sigma(k)$'th row of A' is $e_{i_k}^T$.

A' is non-singular since its rows form a basis of $\mathbb{R}^n$. It remains to show that $AA'^{-1}b = b$. Since $Ax = b$ has a solution and the set R of rows $a_1^T, \ldots, a_r^T$ spans the row space of A, every equation corresponding to a row of $Ax = b$ is a linear combination of the r equations corresponding to the rows in R. Therefore, if x is any vector that satisfies the r equations corresponding to the rows in R then it satisfies all the equations of $Ax = b$. The vector $A'^{-1}b$ satisfies these r equations by the definition of A'. Therefore, $AA'^{-1}b = b$. □

We are now ready to proceed toward the proof of Theorem 3.14. Recall again that we assume that the PPS, $x = P(x)$, is in SNF form, where each equation $x_i = P_i(x)$ is either of the form $x_i = x_j x_k$ (Form$_*$), or is of the form $x_i = \sum_j p_{i,j}x_j + p_{i,0}$ (Form$_+$). There is one equation for each variable. If n is the number of variables, we can assume w.l.o.g. that $|P| \geq 3n$ (i.e. the input has at least 3 bits per variable).

We know that the ratio of largest and smallest non-zero components of $\mathbf{1} - q^*$ is smaller than $2^{2|P|}$ in the strongly connected case (Lemma 3.12). In the general case, two variables may not depend on each other, even indirectly. Nevertheless, we can establish a good upper bound on coordinates of $q^* < 1$. As before, we start with the strongly connected case:

Theorem 3.17. *Given a strongly connected PPS, $x = P(x)$, with $P(\mathbf{1}) = \mathbf{1}$, with LFP q^*, such that $\mathbf{0} < q^* < \mathbf{1}$, and with rational coefficients, then*

$$q_i^* < 1 - 2^{-3|P|}$$

for some $1 \leq i \leq n$.

Proof. Consider the vector $(I - B(\mathbf{1}))(\mathbf{1} - q^*)$. As $P(\mathbf{1}) = \mathbf{1}$, by Lemma 3.3 we have $B(\frac{1}{2}(\mathbf{1} + q^*))(\mathbf{1} - q^*) = \mathbf{1} - q^*$ and so

$$(B(\mathbf{1}) - I)(\mathbf{1} - q^*) = (B(\mathbf{1}) - B(\frac{1}{2}(\mathbf{1} + q^*)))(\mathbf{1} - q^*)$$

This is zero except for coordinates of Form$_*$, as rows of $B(\frac{1}{2}(\mathbf{1} + q^*))$ and $B(\mathbf{1})$ that correspond to Form$_+$ equations are identical. If we have an expression of Form$_*$, $(P(x))_i = x_j x_k$, then

$$\begin{aligned}(B(\mathbf{1}) - I)(\mathbf{1} - q^*)_i &= (B(\mathbf{1}) - B(\frac{1}{2}(\mathbf{1} + q^*)))(\mathbf{1} - q^*)_i \\ &= (1/2)(1 - q_k^*)(1 - q_j^*) + (1/2)(1 - q_j^*)(1 - q_k^*) \\ &= (1 - q_k^*)(1 - q_j^*)\end{aligned}$$

Consequently:

$$\|(I - B(\mathbf{1}))(\mathbf{1} - q^*)\|_\infty \leq \|(\mathbf{1} - q^*)\|_\infty^2 \tag{6}$$

We will distinguish two cases, depending on whether $(I - B(\mathbf{1}))$ is singular or not.

Case 1: $(I - B(\mathbf{1}))$ is non-singular.

In this case, we have that:

$$\mathbf{1} - q^* = (I - B(\mathbf{1}))^{-1}(I - B(\mathbf{1}))(\mathbf{1} - q^*)$$

$$\|\mathbf{1} - q^*\|_\infty \leq \|(I - B(\mathbf{1}))^{-1}\|_\infty \|(I - B(\mathbf{1}))(\mathbf{1} - q^*)\|_\infty$$

$$\|\mathbf{1} - q^*\|_\infty \leq \|(I - B(\mathbf{1}))^{-1}\|_\infty \|(\mathbf{1} - q^*)\|_\infty^2$$

$$\|\mathbf{1} - q^*\|_\infty \geq \frac{1}{\|(I - B(\mathbf{1}))^{-1}\|_\infty} \tag{7}$$

where $\|\cdot\|_\infty$ on matrices is the induced norm of $\|\cdot\|_\infty$ on vectors. $\|A\|_\infty$ for an $n \times m$ matrix A with entries a_{ij} is the maximum absolute value row sum $\max_{i=1}^{n} \sum_{j=1}^{n} |a_{ij}|$.

So an upper bound on $\|(I - B(\mathbf{1}))^{-1}\|_\infty$ will give the lower bound on $\|\mathbf{1} - q^*\|_\infty$ we are looking for. We use Lemma 3.15 to establish such an upper bound. If we take $(I - B(\mathbf{1}))$ to be the matrix A of Lemma 3.15, then noting that the product of all the denominators in $(I - B(\mathbf{1}))$ is at most $2^{|P|}$, Lemma 3.15 gives:

$$\|(I - B(\mathbf{1}))^{-1}\|_\infty \leq n2^{|P|}\|(I - B(\mathbf{1}))\|_\infty^n$$

Of course $\|(I - B(\mathbf{1}))\|_\infty \leq 1 + \|B(\mathbf{1})\|_\infty \leq 3$ (note that here we are using the fact that the system is in SNF normal form). Thus

$$\|(I - B(\mathbf{1}))^{-1}\|_\infty \leq 3^n n2^{|P|}$$

Using inequality (7), and since as discussed, w.l.o.g., $|P| \geq 3n \geq n \log 3 + \log n$, this gives:

$$\|\mathbf{1} - q^*\|_\infty \quad \geq \quad \frac{1}{n} 2^{-|P|} 3^{-n} > 2^{-2|P|}$$

Case 2: $(I - B(\mathbf{1}))$ is singular.

We can look for a small solution v to:

$$(I - B(\mathbf{1}))v = (I - B(\mathbf{1}))(1 - q^*) \tag{8}$$

We will apply Lemma 3.16. We can replace some rows of $(I - B(\mathbf{1}))$ to get the non-singular matrix A' specified in Lemma 3.16, and then use Lemma 3.15 on

$$v' = A'^{-1}(I - B(\mathbf{1}))(1 - q^*)$$

We still have $\|A'\|_\infty \leq 3$, and the product of all the denominators of non-zero entries is smaller than $2^{|P|}$. As for $\|(I - B(1))^{-1}\|_\infty$ before:

$$\|A'^{-1}\|_\infty \leq 3^n n2^{|P|}$$

Using inequality (6), we have

$$\|v'\|_\infty \leq 3^n n2^{|P|}\|(\mathbf{1} - q^*)\|_\infty^2 \tag{9}$$

By equation (8), we have that $(I - B(1))((1 - q^*) - v') = 0$. Thus $(1 - q^*) - v'$ is an eigenvector of $B(\mathbf{1})$ with eigenvalue 1. But we know that $B(\mathbf{1})$ is nonnegative, irreducible, and has spectral radius bigger than 1 (because $q^* < \mathbf{1}$ by assumption, see e.g., [24] proof of Theorem 8.1). Thus Perron-Frobenius theory (e.g., see Corollary 8.1.29 in [37]) gives us that $(1-q^*)-v'_i$ is not a positive vector (because the only positive eigenvectors are associated with the top eigenvalue). Thus some coordinate i has:

$$v'_i \geq 1 - q^*_i$$

Thus, by inequality (9), we have:

$$1 - q^*_i \leq 3^n n2^{|P|}\|(\mathbf{1} - q^*)\|_\infty^2$$

but the proof of Lemma 3.12 gave that:

$$(1 - q_i^*)2^{|P|+n} \geq \|(\mathbf{1} - q^*)\|_\infty$$

Combining these inequalities, we have

$$\begin{aligned} 1 - q_i^* &\leq 3^n n 2^{|P|} \|(\mathbf{1} - q^*)\|_\infty^2 \\ &\leq 3^n n 2^{|P|} (1 - q_i^*) 2^{|P|+n} \|(\mathbf{1} - q^*)\|_\infty \end{aligned}$$

Dividing both sides by $(1 - q_i^*)$, we have that:

$$\begin{aligned} \|(\mathbf{1} - q^*)\|_\infty &\geq \frac{1}{6^n n 2^{2|P|}} \\ &> 2^{-3|P|} \end{aligned}$$

□

We are almost ready to prove Theorem 3.14 for general (not necessarily strongly connected) PPSs. We first need the following lemma:

Lemma 3.18. *Any variable x_i either depends (directly or indirectly)[6] on a variable in a bottom SCC S such that $P_S(\mathbf{1}) = \mathbf{1}$ (meaning there is no "non-proper" variable in that SCC), or it depends (directly or indirectly) on some variable x_j of Form$_+$, with $P(x)_j = p_{j,0} + \sum_{k=1}^n p_{j,k} x_k$ where $\sum_{k=0}^n p_{i,j} < 1$ (thus, x_j is a non-proper variable).*

Proof. Let D_i denote the set of variables that x_i depends on (including x_i itself). Suppose that every variable of Form$_+$, x_j in D_i, with $P(x)_j = p_{j,0} + \sum_{k=1}^n p_{j,k} x_k$ has $\sum_{k=0}^n p_{j,k} = 1$. Then we can verify that $P_{D_i}(\mathbf{1}) = \mathbf{1}$. D_i contains some bottom SCC $S \subseteq D_i$. For this SCC $P_S(\mathbf{1}) = \mathbf{1}$. □

Proof of Theorem 3.14. Recall, we wish to show that for a general PPS, $x = P(x)$, in SNF normal form, with n variables, with rational coefficients, and with LFP, $\mathbf{0} < q^* < \mathbf{1}$, we have $q_i^* < 1 - 2^{-4|P|}$, for all $1 \leq i \leq n$.

Suppose that variable x_j is of Form$_+$ with $P(x)_j = p_{j,0} + \sum_{k=1}^n p_{j,k} x_k$ where $\sum_{k=0}^n p_{j,k} < 1$. Then $q_j^* = P(q^*)_j$ has $q_j^* \leq \sum_{k=0}^n p_{j,k}$. $1 - \sum_{k=0}^n p_{j,k}$ is a rational with a denominator smaller than the product of the denominators of all the $p_{j,k}$. We have:

$$1 - \sum_{k=0}^{n} p_{j,k} \geq 2^{-|P|}$$

Thus in such a case:

$$q_j^* \leq 1 - 2^{-|P|} \ .$$

Lemma 3.18 says that any x_i either depends on such a variable, or on a variable to which Theorem 3.17 applies. That is, x_i depends on some x_j with

$$q_j^* \leq 1 - 2^{-3|P|} \ .$$

[6] meaning that in the dependency graph the node for x_i can reach the node for the other variable.

Therefore, for any i, unless already $q_i^* \leq 1-2^{-3|P|}$, there is some sequence $x_{l_1}, x_{l_2}, \ldots, x_{l_m}$ of *distinct* variables with $l_1 = i$ and $l_m = j$, and such that for every k, $0 \leq k < m$, $P_{l_k}(x)$ contains a term with $x_{l_{k+1}}$ in it (e.g., take the shortest path in the dependency graph from x_i to x_j). If x_{l_k} has Form$_*$, then $q^*_{l_k} \leq q^*_{l_{k+1}}$. If x_{l_k} has Form$_+$, then $q^*_{l_k} \leq p_{l_k,l_{k+1}} q^*_{l_{k+1}} + (1 - p_{l_k,l_{k+1}}) = 1 - p_{l_k,l_{k+1}}(1 - q^*_{l_{k+1}})$. Thus $(1 - q^*_{l_k}) \geq p_{l_k,l_{k+1}}(1 - q^*_{l_{k+1}})$. Thus, by an easy induction:

$$1 - q_i^* \geq (\prod_{x_{l_k} \text{ has Form}_+} p_{l_k,l_{k+1}})(1 - q_j^*) \ .$$

Again, $|P|$ is at least the number of bits describing all these rationals $p_{l_k,l_{k+1}}$ (which are all part of the input), and thus

$$1 - q_i^* \geq 2^{-|P|}(1 - q_j^*) \ .$$

Since we already know that $q_j^* \leq 1 - 2^{-3|P|}$, i.e., that $(1 - q_j^*) \geq 2^{-3|P|}$, we obtain:

$$1 - q_i^* \geq 2^{-|P|} 2^{-3|P|} = 2^{-4|P|}$$

This completes the proof of Theorem 3.14. □

4 Polynomial time in the standard Turing model of computation

In Section 3 (Theorem 3.2) we have shown that for a PPS, $x = P(x)$, using $(4|P| + j)$ iterations of Newton's method starting at $x^{(0)} := \mathbf{0}$, we obtain q^* within additive error 2^{-j}. However, performing even $|P|$ iterations of Newton's method *exactly* may not be feasible in P-time in the *Turing* model, because the encoding size of iterates $x^{(k)}$ can become very large. Specifically, by repeated squaring, the rational numbers representing the iterate $x^{(|P|)}$ may require encoding size exponential in $|P|$.

In this section, we show that we can nevertheless approximate in P-time the LFP q^* of a PPS, $x = P(x)$. We do so by showing that we can *round down* all coordinates of each Newton iterate $x^{(k)}$ to a suitable polynomial length, and still have a well-defined iteration that converges in nearly the same number of iterations to q^*. Throughout this section we assume every PPS is in SNF form.

Definition 4.1. *("Rounded down Newton's method", with rounding parameter h.) Given a PPS, $x = P(x)$, with LFP q^*, where $\mathbf{0} < q^* < \mathbf{1}$, in the "rounded down Newton's method" with integer rounding parameter $h > 0$, we compute a sequence of iteration vectors $x^{[k]}$, where the initial starting vector is again $x^{[0]} := \mathbf{0}$, and such that for each $k \geq 0$, given $x^{[k]}$, we compute $x^{[k+1]}$ as follows:*

1. *First, compute $x^{\{k+1\}} := \mathcal{N}_P(x^{[k]})$, where the Newton iteration operator $\mathcal{N}_P(x)$ was defined in equation (2). (Of course we need to show that all such Newton iterations are defined.)*
2. *For each coordinate $i = 1, \ldots, n$, set $x_i^{[k+1]}$ to be equal to the maximum (non-negative) multiple of 2^{-h} which is $\leq \max(x_i^{\{k+1\}}, 0)$. (In other words, round down $x^{\{k+1\}}$ to the nearest multiple of 2^{-h}, while making sure that the result is non-negative.)*

Theorem 4.2 (Main Theorem of Section 4). *Given a PPS, $x = P(x)$, with LFP q^*, such that $0 < q^* < 1$, if we use the rounded down Newton's method with parameter $h = j + 2 + 4|P|$, then the iterations are all defined, for every $k \geq 0$ we have $0 \leq x^{[k]} \leq q^*$, and furthermore after $h = j + 2 + 4|P|$ iterations we have: $\|q^* - x^{[j+2+4|P|]}\|_\infty \leq 2^{-j}$.*

We prove this via some lemmas. The first lemma proves that the iterations are always well-defined, and yield vectors $x^{[k]}$ such that $\mathbf{0} \leq x^{[k]} \leq q^*$. Note however that, unlike Newton iteration using exact arithmetic, we *do not* claim (as in Proposition 3.1) that $x^{[k]}$ converges *monotonically* to q^*. It may not. It turns out we don't need this: all we need is that $0 \leq x^{[k]} \leq q^*$, for all k. In particular, it may not hold that $P(x^{[k]}) \geq x^{[k]}$. For establishing the monotone convergence of Newton's method on MPSs (Proposition 3.1), the fact that $P(x^{(k)}) \geq x^{(k)}$ is key (see [24]). Indeed, note that for PPSs, once we know that $(P(x^{(k)}) - x^{(k)}) \geq 0$, Theorem 3.7 and the defining equation of Newton iteration, (1), already proves monotone convergence: $x^{(k)}$ is well-defined and $x^{(k+1)} \geq x^{(k)} \geq 0$, for all k. However, $P(x^{[k]}) \geq x^{[k]}$ may no longer hold after rounding down. If, for instance, the polynomial $P_i(x)$ has degree 1 (i.e., has Form_+), then one can show that after any positive number of iterations $k \geq 1$, we will have that $P_i(x^{\{k\}}) = x_i^{\{k\}}$. So, if we are unlucky, rounding down each coordinate of $x^{\{k\}}$ to a multiple of 2^{-h} could indeed give $(P(x^{[k+1]}))_i < x_i^{[k+1]}$.

Lemma 4.3. *If we run the rounded down Newton method starting with $x^{[0]} := \mathbf{0}$ on a PPS, $x = P(x)$, with LFP q^*, $\mathbf{0} < q^* < \mathbf{1}$, then for all $k \geq 0$, $x^{[k]}$ is well-defined and $0 \leq x^{[k]} \leq q^*$.*

Proof. We prove this by induction on k. The base case $x^{[0]} = 0$ is immediate. Suppose the claim holds for k and thus $0 \leq x^{[k]} \leq q^*$. Lemma 3.4 tells us that

$$q^* - x^{\{k+1\}} = (I - B(x^{[k]}))^{-1} \frac{B(q^*) - B(x^{[k]})}{2}(q^* - x^{[k]})$$

Now the fact that $0 \leq x^{[k]} \leq q^*$ yields that each of the following inequalities hold: $(q^* - x^{[k]}) \geq 0$, $B(q^*) - B(x^{[k]}) \geq 0$. Furthermore, by Theorem 3.7, we have that $\rho(B(x^{[k]})) < 1$, and thus that $(I - B(x^{[k]}))$ is non-singular and $(I - B(x^{[k]}))^{-1} \geq 0$. We thus conclude that $q^* - x^{\{k+1\}} \geq 0$, i.e., that $x^{\{k\}} \leq q^*$. The rounding down ensures that $0 \leq x_i^{[k+1]} \leq x_i^{\{k+1\}}$ unless $x_i^{\{k+1\}} < 0$, in which case $x_i^{[k+1]} = 0$. In both cases, we have that $0 \leq x^{[k+1]} \leq q^*$. So we are done by induction. □

The next key lemma shows that the rounded version still makes good progress towards the LFP.

Lemma 4.4. *For a PPS, $x = P(x)$, with LFP q^*, such that $\mathbf{0} < q^* < \mathbf{1}$, if we apply the rounded down Newton's method with parameter h, starting at $x^{[0]} := \mathbf{0}$, then for all $k \geq 0$, we have:*

$$\|q^* - x^{[k]}\|_\infty \leq (2^{-k} + 2^{-h+1}) \cdot 2^{4|P|}$$

Proof. Since $x^{[0]} := 0$:

$$q^* - x^{[0]} = q^* \leq \mathbf{1} \leq \frac{1}{(\mathbf{1} - q^*)_{\min}}(\mathbf{1} - q^*) \tag{10}$$

For any $k > 0$, if $q^* - x^{[k-1]} \leq \lambda(\mathbf{1} - q^*)$, then by Lemma 3.9 we have:

$$q^* - x^{\{k\}} \leq (\frac{\lambda}{2})(\mathbf{1} - q^*) \tag{11}$$

Observe that after every iteration $k > 0$, in every coordinate i we have:

$$x_i^{[k]} \geq x_i^{\{k\}} - 2^{-h} \tag{12}$$

This holds simply because we are rounding down $x_i^{\{k\}}$ by at most 2^{-h}, unless it is negative in which case $x_i^{[k]} = 0 > x_i^{\{k\}}$. Combining the two inequalities (11) and (12) yields the following inequality:

$$q^* - x^{[k]} \leq (\frac{\lambda}{2})(\mathbf{1} - q^*) + 2^{-h}\mathbf{1} \leq (\frac{\lambda}{2} + \frac{2^{-h}}{(\mathbf{1} - q^*)_{\min}})(\mathbf{1} - q^*)$$

Taking inequality (10) as the base case (with $\lambda = \frac{1}{(\mathbf{1}-q^*)_{\min}}$), by induction on k, for all $k \geq 0$:

$$q^* - x^{[k]} \leq (2^{-k} + \sum_{i=0}^{k-1} 2^{-(h+i)}) \frac{1}{(\mathbf{1} - q^*)_{\min}} (\mathbf{1} - q^*)$$

But $\sum_{i=0}^{k-1} 2^{-(h+i)} \leq 2^{-h+1}$ and $\frac{\|\mathbf{1}-q^*\|_\infty}{(\mathbf{1}-q^*)_{\min}} \leq \frac{1}{(\mathbf{1}-q^*)_{\min}} \leq 2^{4|P|}$, by Theorem 3.14. Thus:

$$q^* - x^{[k]} \leq (2^{-k} + 2^{-h+1})2^{4|P|}\mathbf{1}$$

Clearly, we have $q^* - x^{[k]} \geq 0$ for all k. Thus we have shown that for all $k \geq 0$:

$$\|q^* - x^{[k]}\|_\infty \leq (2^{-k} + 2^{-h+1})2^{4|P|}.$$

□

We can now show the main theorem:

*Proof of Theorem 4.2 (**Main Theorem of Sec. 4**).* In Lemma 4.4 let $k := h := j + 2 + 4|P|$. We have: $\|q^* - x^{[j+2+4|P|]}\|_\infty \leq 2^{-(j+2)} + 2^{-(j+1)} \leq 2^{-j}$. □

Corollary 4.5. *Given any PPS, $x = P(x)$, with LFP q^*, we can approximate q^* within additive error 2^{-j} in time polynomial in $|P|$ and j (in the standard Turing model of computation). More precisely, we can compute a vector v, $\mathbf{0} \leq v \leq q^*$, such that $\|q^* - v\|_\infty \leq 1/2^{-j}$.*

The same results hold for the computation of the extinction probabilities of a multitype branching process, the termination probabilities of a SCFG, and of a 1-RMC.

Proof. Firstly, by Propositions 2.1 and 2.2, we can assume $x = P(x)$ is in SNF form, and that $\mathbf{0} < q^* < \mathbf{1}$. By Theorem 4.2, the rounded down Newton's method with parameter $h = j+2+4|P|$, for $h = j + 2 + 4|P|$ iterations, computes a rational vector $v = x^{[h]}$ such that $v \in [0, 1]^n$, and $\|q^* - v\|_\infty \leq 1/2^{-h}$.

Furthermore, for all k, with $0 \leq k \leq h$, $x^{[k]}$ has encoding size polynomial in $|P|$ and j. We then simply need to note that all the linear algebra operations, that is: matrix multiplication, addition, and matrix inversion, required in a single iteration of Newton's method, can be performed exactly on rational inputs in polynomial time and yield rational results with a polynomial size. □

5 Norm bounds for the matrix $(I - B(q^*))^{-1}$

In order to establish results about quadratic convergence of Newton's method for PPSs, and for the quantitative decision problem for the LFP of PPSs, we need a bound on the norm of the matrix $(I - B(q^*))^{-1}$ when $q^* < 1$, which is what we establish in this section.[7]

[7]Let us additionally mention that these norm bounds play an important role in our subsequent work, on algorithms for Branching Markov Decision Processes [18]. Indeed, these norm bounds first appeared in (the preprint arXiv version of) [18].

We use the $\|.\|_\infty$ matrix norm (which, recall, is the maximum absolute value row sum). For a PPS, $x = P(x)$ with n variables, recall that its variable *dependency graph* is defined to be the digraph $H = (V, E)$, with vertices $V = \{x_1, \ldots, x_n\}$, such that $(x_i, x_j) \in E$ if and only if in $P_i(x) \equiv \sum_{r \in R_i} p_r x^{v(\alpha_r)}$ there is a coefficient $p_r > 0$ such that $v(\alpha_r)_j > 0$. Intuitively, $(x_i, x_j) \in E$ means that x_i "depends directly" on x_j. Recall that an MPS or PPS, $x = P(x)$, is strongly connected if its dependency graph H is strongly connected.

The aim of this section is to prove the following Theorem:

Theorem 5.1. *If $x = P(x)$ is a PPS in SNF form with LFP $q^* > \mathbf{0}$, then*
(i) If $q^* < \mathbf{1}$ and $\mathbf{0} \leq y < \mathbf{1}$, then $(I - B(\frac{1}{2}(y + q^*)))^{-1}$ exists and is non-negative, and

$$\|(I - B(\frac{1}{2}(y + q^*)))^{-1}\|_\infty \leq 2^{10|P|} max\ \{2(\mathbf{1} - y)^{-1}_{\min}, 2^{|P|}\}$$

(ii) If $q^* = \mathbf{1}$ and $x = P(x)$ is strongly connected (i.e. every variable depends directly or indirectly on every other) and $\mathbf{0} \leq y < \mathbf{1} = q^*$, then $(I - B(y))^{-1}$ exists and is non-negative, and

$$\|(I - B(y))^{-1}\|_\infty \leq 2^{4|P|} \frac{1}{(\mathbf{1} - y)_{\min}}$$

Before proving this Theorem, we shall need to develop some lemmas. The first lemma relates the dependency between variables to the nonzero entries of powers of the matrix $B(\mathbf{1})$ and provides a lower bound for such entries.

Lemma 5.2. *Given a PPS $x = P(x)$ in SNF form, and variables x_i,x_j:*

(i) If x_i depends on x_j then there is a positive integer k, with $1 \leq k \leq n$, such that

$$(B(\mathbf{1})^k)_{ij} \geq 2^{-|P|}$$

(ii) If $(B(\mathbf{1})^k)_{ij} > 0$ for some positive integer k, with $1 \leq k \leq n$, then x_i depends on x_j.

(iii) If x_i depends on x_j "only via variables of Form$_+$", i.e., if there is a path $x_{l_1}, \ldots, x_{l_m}$ in the dependency graph such that $l_1 = i$ and $l_m = j$, and such that for each $1 \leq h \leq m - 1$, $x_{l_h} = P_{l_h}(x) = p_{l_h,0} + \sum_{g=1}^{n} p_{l_h,g} x_g$ has Form$_+$ with $p_{l_h,l_{h+1}} > 0$, then there is a $1 \leq k \leq n$ such that, for any vector x, such that $\mathbf{0} \leq x \leq \mathbf{1}$,

$$(B(x)^k)_{ij} \geq 2^{-|P|}$$

Proof.

(i) Let the sequence of variables $x_{l_1}, \ldots, x_{l_k}$ constitute a shortest path from x_i and x_j, such that $k \geq 2$. Such a shortest path exists, since x_i depends on x_j. So $x_i = x_{l_1}$, and $x_j = x_{l_k}$, and $x_{l_{h+1}}$ appears in the expression for $P_{l_h}(x)$, and $1 \leq h \leq k - 1$. Note that we must have $k \leq n$. Thus $(B(1))_{l_h l_{h+1}} > 0$ for $1 \leq h \leq k - 1$. But note that since $B(1)$ is a non-negative matrix, $(B(1)^{k-1})_{ij} \geq \prod_{h=1}^{k-1} (B(1))_{l_h l_{h+1}}$. Since we have chosen a shortest (non-empty) path from x_i to x_j, and since $x = P(x)$ is in SNF form, each $(B(1))_{l_h l_{h+1}}$ that is not exactly 1 must be a distinct rational coefficient in P, not appearing elsewhere along the path, and thus $\prod_{h=1}^{k-1} (B(1))_{l_h l_{h+1}} \geq 2^{-|P|}$.

(ii) For $k \geq 1$, we can expand $(B(1)^k)_{ij}$ into a sum of n^{k-1} terms of the form $\prod_{h=1}^{k}(B(1))_{l_h l_{h+1}}$ with $l_1 = i, l_{k+1} = j$ and $(l_2, \ldots, l_k) \in \{1, \ldots, n\}^{k-1}$. At least one of these has $\prod_{h=1}^{k}(B(1))_{l_h l_{h+1}} > 0$. In that case, $x_{h_1}, \ldots, x_{h_{k+1}}$ is a path in the dependency graph starting at x_i and ending at x_j.

(iii) Let us choose $x_{l_1}, \ldots, x_{l_k}$ to be a shortest path from x_i to x_j, with $k \geq 2$, and such that every equation $x_{l_h} = P_{l_h}(x)$ along the path, for all $h \in \{1, \ldots, k-1\}$ has Form_+. Clearly, we must have $k \leq n$. By monotonicity of $B(z)$ in $z \geq 0$, we have $(B(1)^{k-1})_{ij} \geq B(x)^{k-1}$. Furthermore, since $x_{l_1}, \ldots, x_{l_k}$ is a path from x_i to x_j, we have $(B(x))^{k-1}_{i,j} \geq \prod_{h=1}^{k-1}(B(x))_{l_h l_{h+1}}$. Moreover, since each equation $x_{l_h} = P(x)_{l_h}$ has Form_+, for every $h \in \{1, \ldots, k-1\}$, we must have $(B(x))_{l_h l_{h+1}} = (B(1))_{l_h l_{h+1}}$ (because all the partial derivatives of linear expressions are constants). But we argued in (i) that, when $x_{l_1}, \ldots, x_{l_k}$ constitutes a shortest path from x_i to x_j, $\prod_{h=1}^{k-1}(B(1))_{l_h l_{h+1}} \geq 2^{-|P|}$.

□

We need a basic result from the Perron-Frobenius theory of non-negative matrices. We are not aware of a source that contains a statement exactly equivalent to (or implying) the following Lemma, so we shall provide a proof. However it is entirely possible (and likely) that such a Lemma has appeared elsewhere. Lemma 19 of [23] provides a similar result for the case when the matrix A is irreducible.

Lemma 5.3. *If A is a non-negative matrix, and vector $u > 0$ is such that $Au \leq u$ and $\|u\|_\infty \leq 1$, and $\alpha, \beta \in (0, 1)$ are constants such that for every $i \in \{1, \ldots, n\}$, one of the following two conditions holds:*

(I) $(Au)_i \leq (1 - \beta)u_i$, or

(II) there is some k, $1 \leq k \leq n$, and some j, such that $(A^k)_{ij} \geq \alpha$ and $(Au)_j \leq (1 - \beta)u_j$

then $\rho(A) < 1$, $(I - A)$ is non-singular, and

$$\|(I - A)^{-1}\|_\infty \leq \frac{n}{u_{\min}^2 \alpha \beta}$$

Proof. First, suppose that some $i \in \{1, \ldots, n\}$, satisfies condition (I). Then, we claim that it satisfies condition (II), except that we must take $k = 0$. Specifically, if we let $k = 0$, then since $A^0 = I$, and $(A^0)_{ii} = I_{ii} = 1 \geq \alpha$, condition (II) boils down to $(Au)_i \leq (1 - \beta)u_i$. So, to prove the statement, it suffices to only consider condition (II) but to allow $k = 0$ in that condition.

So, by assumption, given any $i \in \{1, \ldots, n\}$, there is some $0 \leq k \leq n$ and some j, such that

$$(A^k)_{ij} \geq \alpha > 0 \tag{13}$$

and moreover $(Au)_j \leq (1 - \beta)u_j$, which we can rewrite as:

$$u_j - (Au)_j \geq \beta u_j \quad (> 0) \tag{14}$$

Let $u_{\min} = \min_i u_i$. We thus have that for every i:

$$\begin{aligned}
(A^n u)_i &= (u - \sum_{l=0}^{n-1} A^l(u - Au))_i \\
&\leq (u - A^k(u - Au))_i \qquad \text{(because } A^l \geq 0 \text{ and } (u - Au) \geq 0) \\
&= u_i - \sum_{j'=1}^{n} A^k_{ij'}(u_{j'} - (Au)_{j'}) \\
&\leq u_i - A^k_{ij}(u_j - (Au)_j) \quad \text{(again, because } A^k_{i,j'} \geq 0 \text{ and } (u_{j'} - (Au)_{j'}) \geq 0 \text{ for every } j') \\
&\leq u_i - \alpha\beta u_j \qquad \text{(by (13) and (14))} \\
&\leq u_i - \alpha\beta u_{\min} \\
&\leq u_i - u_{\min}\alpha\beta u_i \qquad \text{(recalling that by assumption } \|u\|_\infty \leq 1)
\end{aligned}$$

We have that $A^n u \leq (1 - u_{\min}\alpha\beta)u$. Of course $(1 - u_{\min}\alpha\beta) < 1$. So we have that

$$A^{mn}u \leq (1 - u_{\min}\alpha\beta)^m u$$

For any integer $d \geq 0$, $A^d u \leq u$. Thus also, for every $d \geq 0$,

$$A^d u \leq (1 - u_{\min}\alpha\beta)^{\lfloor \frac{d}{n} \rfloor} u \tag{15}$$

We thus have that, as $m \to \infty$, $A^m u \to 0$. Since $u > 0$ and $A \geq 0$, this implies that as $m \to \infty$, $A^m \to 0$ (coordinate-wise), or in other words that $\lim_{m\to\infty} \|A^m\|_\infty = 0$. This is equivalent to saying that the spectral radius $\rho(A) < 1$. This implies that the inverse matrix $(I - A)^{-1} = \sum_{k=0}^{\infty} A^k \geq 0$ exists, by Lemma 3.6.
We will use the following easy fact:

Lemma 5.4. *If M is a nonnegative $n \times n$ matrix, $u > 0$ is a vector with $\|u\|_\infty \leq 1$, and $\lambda > 0$ is a real number satisfying $Mu \leq \lambda u$ then*

$$\|M\|_\infty \leq \frac{\lambda}{u_{\min}}$$

Proof. Since M is non-negative, $\|M\|_\infty$ is the maximum row sum of M. There is thus an i such that

$$\|M\|_\infty = \sum_j m_{ij}$$

where $m_{i,j}$ are the entries of M. For this i:

$$\begin{aligned}
\lambda u_i &\geq (Mu)_i \\
&= \sum_j m_{ij} u_j \\
&\geq \sum_j m_{ij} u_{\min} \\
&= \|M\|_\infty u_{\min}
\end{aligned}$$

but $u_i \leq 1$ giving us $||M||_\infty \leq \frac{\lambda}{u_{\min}}$. □

Now we can complete the proof of Lemma 5.3:

$$\begin{aligned}(I-A)^{-1}u = (\sum_{k=0}^{\infty} A^k)u &= \sum_{k=0}^{\infty} A^k u \\ &\leq \sum_{k=0}^{\infty} (1-u_{\min}\alpha\beta)^{\lfloor \frac{k}{n} \rfloor} u \qquad \text{(by (15))} \\ &= (\sum_{m=0}^{\infty} n(1-u_{\min}\alpha\beta)^m u \\ &= n\frac{1}{u_{\min}\alpha\beta}u\end{aligned}$$

the last equality holding because the geometric series sum gives $\sum_{m=0}^{\infty}(1-u_{\min}\alpha\beta)^m = \frac{1}{u_{\min}\alpha\beta}$. Lemma 5.4, with $M := (I-A)^{-1} = \sum_{k=0}^{\infty} A^k$, and $\lambda := n\frac{1}{u_{\min}\alpha\beta}$, now yields:

$$\|(I-A)^{-1}\|_\infty \leq n\frac{1}{u_{\min}^2\alpha\beta}$$

and this completes the proof of Lemma 5.3. □

We note also the following easy consequence of the assumption that the LFP $q^* > \mathbf{0}$.

Proposition 5.5. *For a PPS, $x = P(x)$, with LFP $q^* > \mathbf{0}$, for every variable x_i either $P_i(0) > 0$ or x_i depends on a variable x_j with $P_j(0) > 0$.*

Proof. Suppose, for contradiction, that a variable x_i has $P_i(0) = 0$ and depends only on variables x_j which have $P_j(0) = 0$. Then $P_i^n(0) = 0$ for all n. But $P^n(0) \to q^*$ as $n \to \infty$ (see, e.g., Theorem 3.1 from [24]). So $q_i^* = 0$. □

To prove Theorem 5.1, we treat first the case of a PPS when all the equations, $x_i = P_i(x)$, are linear. In this case, the Jacobian matrix $B(x)$ is a constant matrix B, independent of x. The following lemma implies Theorem 5.1 for the case of a purely linear PPS.

Lemma 5.6. *Let $x = P(x)$ be a PPS in SNF form that has no equations of Form$*$, and has LFP $q^* > \mathbf{0}$, and let B be the constant Jacobian matrix of $P(x)$, (i.e., $B = B(x)$ for all x). Then $\rho(B) < 1$, $(I-B)^{-1}$ exists, is nonnegative, and*

$$\|(I-B)^{-1}\|_\infty \leq n2^{2|P|}.$$

Proof. First, note that B is a sub-stochastic matrix i.e. $B\mathbf{1} \leq \mathbf{1}$. Let T be the set of variables $T = \{x_i | (B\mathbf{1})_i < 1\}$. Since $P_i(x) \equiv \sum_{i=1}^n p_{i,j}x_j + p_{i,0}$, this means that $x_i \in T$ iff $(B1)_i = \sum_{j=1}^n \frac{\partial P_i(x)}{\partial x_j} = \sum_{j=1}^n p_{i,j} < 1$. Note that if $P_i(0) > 0$ then $p_{i,0} > 0$, hence $\sum_{j=1}^n p_{i,j} < 1$ and thus $x_i \in T$. Since $q^* > \mathbf{0}$, it follows from Proposition 5.5, that for every variable x_i, either x_i itself is in T, or x_i depends (possibly indirectly) on a variable $x_j \in T$.

Since the entries of B are either 0, 1, or coefficients $p_{i,j}$ from $P(x)$, we see that for every variable $x_i \in T$, we have that $(B\mathbf{1})_i = \sum_{j=1}^{n} p_{i,j} \leq (1 - 2^{-|P|})$ holds.[8]

For any variable $x_r \notin T$, there is a variable $x_i \in T$ that x_r depends on. All the variables in the PPS are of Form$_+$. Thus, by Lemma 5.2 (iii), there is a k, $1 \leq k \leq n$, such that $((B)^k)_{ri} \geq 2^{-|P|}$.

We can thus apply Lemma 5.3 with matrix $A := B$ and vector $u := \mathbf{1}$, with $\alpha := \beta := 2^{-|P|}$, because we have just established that condition (I) of that Lemma applies to variables in T, and condition (II) of that Lemma applies to variables that are not in T. Thus Lemma 5.3 gives us that $\rho(B) < 1$, $(I - B)^{-1}$ exists, is nonnegative, and $\|(I - B)^{-1}\|_\infty \leq (\frac{1}{\mathbf{1}_{\min}})^2 n 2^{2|P|}$. Of course, $\mathbf{1}_{\min} = 1$. □

We are now ready to prove parts ***(i)*** and ***(ii)*** of Theorem 5.1 for general PPS.

Proof of Theorem 5.1, Part (i).
When $q^* < \mathbf{1}$, we can say something stronger than Proposition 5.5.

Lemma 5.7. *For any PPS, x=P(x), in SNF form, with LFP $\mathbf{0} < q^* < \mathbf{1}$, for any variable x_i, either:*

(I) the equation $x_i = P_i(x)$ has Form$_*$, or else $P_i(\mathbf{1}) < 1$. Or,

(II) x_i depends (directly or indirectly) on a variable x_j, such that $x_j = P_j(x)$ is of Form$_*$, or else $P_j(\mathbf{1}) < 1$.

Proof. Suppose, for contradiction, that there is a variable x_i for which neither (I) nor (II) holds. Let D_i be the set of variables that x_i depends on, unioned together with $\{x_i\}$ itself. For any vector x, consider the subvector x_{D_i}, which consists of the components of x with coordinates in D_i. We can consider the subset of the equations $x_{D_i} = P_{D_i}(x)$. By transitivity of dependency, $P_{D_i}(x)$ contains only terms in the variables x_{D_i}. So $x_{D_i} = P_{D_i}(x) = P_{D_i}(x_{D_i})$ is itself a PPS. Since by assumption neither (I) nor (II) hold for x_i, we have that $x_{D_i} = P_{D_i}(x_{D_i})$ contains no equations of Form$_*$ and $P_{D_i}(\mathbf{1}) = \mathbf{1}$. Since, therefore, $P_{D_i}(x_{D_i})$ is linear, we can rewrite $x_{D_i} = P_{D_i}(x_{D_i})$ as $x_{D_i} = B_{D_i}x_{D_i} + P_{D_i}(0)$ and hence $(I - B_{D_i})x_{D_i} = P_{D_i}(0)$. Lemma 5.6 applied to the PPS $x_{D_i} = P_{D_i}(x_{D_i})$ gives us that, in particular, $(I - B_{D_i})$ is non-singular. Consequently $x_{D_i} = P_{D_i}(x_{D_i})$ has a unique solution. But we already said that $\mathbf{1}$ is a solution, $P_{D_i}(\mathbf{1}) = \mathbf{1}$, and so $q^*_{D_i} = \mathbf{1}$. This contradicts $q^* < \mathbf{1}$. So there can be no x_i for which neither (I) nor (II) holds. □

To obtain the conclusion of case ***(i)*** of Theorem 5.1, assuming all of the premises of the Theorem's statement, we will now aim to use Lemma 5.3, applied to $A := B(\frac{1}{2}(y + q^*))$, and $u := \mathbf{1} - q^*$. We need to show that the conditions of the lemma hold for this matrix A and vector u. First, it is clear that $\mathbf{1} - q^* > \mathbf{0}$ and $\|\mathbf{1} - q^*\|_\infty \leq 1$ since $\mathbf{0} \leq q^* < \mathbf{1}$. Second, the condition that $B(\frac{1}{2}(y + q^*)))(\mathbf{1} - q^*) \leq (\mathbf{1} - q^*)$ follows from Lemma 3.5: since $\mathbf{0} \leq y < \mathbf{1}$, it follows by monotonicity of $B(z)$ in z that $B(\frac{1}{2}(y + q^*)))(\mathbf{1} - q^*) \leq (\mathbf{1} - q^*)$.

It remains to establish that for every $i \in \{1, \ldots, n\}$, condition (I) or (II) of Lemma 5.3 holds for some $\alpha, \beta \in (0, 1)$. By Lemma 5.7, every variable x_i either depends on a variable, or is itself equal to a variable, x_j, such that $x_j = P_j(x)$ is of Form$_*$ or $P_j(1) < 1$. We can clearly assume that

[8] This inequality holds because we assume each positive input probability $p_{i,j}$ is represented as a ratio $\frac{a_j}{b_j}$ of positive integers in the encoding of $x = P(x)$, and thus $1 - \sum_{j=1}^{n} \frac{a_j}{b_j}$ can be represented as a ratio $\frac{a}{b}$ of two positive integers where the denominator is $b = \prod_{j=1}^{n} b_j$. But then $(1 - \sum_{j=1}^{n} \frac{a_j}{b_j}) = \frac{a}{b} \geq 1/\prod_{j=1}^{n} b_j \geq \frac{1}{2^{|P|}}$.

such a dependence is via a path consisting only of variables of Form$_+$, in the sense of Lemma 5.2 (iii), and thus for any x_i there is a $0 \leq k \leq n$ with $(B^k(\frac{1}{2}(y+q^*)))_{ij} \geq 2^{-|P|}$, for some x_j with either $x_j = P_j(x)$ of Form$_*$ or $P_j(1) < 1$. Thus, it suffices to show that for such an x_j we have $((B(\frac{1}{2}(y+q^*))(\mathbf{1}-q^*))_j \leq (1-\beta)(\mathbf{1}-q^*)_j$ for some $\beta > 0$.

For any variable x_j such that $x_j = P_j(x)$ has Form$_*$, we have that $x_j = x_k x_l$ for some variables k and l. Thus, since $\frac{\partial P_j(x)}{\partial x_k} = x_l$ and $\frac{\partial P_j(x)}{\partial x_l} = x_k$, we have that:

$$\begin{aligned}
& (B(\frac{1}{2}(q^*+y))(\mathbf{1}-q^*))_j \\
= \;& \frac{1}{2}(q_k^*+y_k)(1-q_l^*) + \frac{1}{2}(q_l^*+y_l)(1-q_k^*) \\
= \;& \frac{1}{2}((q_k^*+1)-(1-y_k))(1-q_l^*) + \frac{1}{2}((q_l^*+1)-(1-y_l))(1-q_k^*) \\
= \;& \frac{1}{2}((q_k^*+1)(1-q_l^*) - (1-y_k)(1-q_l^*) + (q_l^*+1)(1-q_k^*) - (1-y_l)(1-q_k^*)) \\
= \;& \frac{1}{2}(2 - 2q_k^*q_l^* - (1-y_l)(1-q_k^*) - (1-y_k)(1-q_l^*)) \\
\leq \;& \frac{1}{2}(2 - 2q_k^*q_l^* - (\mathbf{1}-y)_{\min}((1-q_k^*)+(1-q_l^*))) \\
\leq \;& \frac{1}{2}(2 - 2q_k^*q_l^* - (\mathbf{1}-y)_{\min}((1-q_k^*)+(1-q_l^*)-(1-q_k^*)(1-q_l^*))) \\
= \;& (1-q_j^*) - \frac{1}{2}(\mathbf{1}-y)_{\min}(1-q_j^*) \\
= \;& (1-\frac{1}{2}(\mathbf{1}-y)_{\min})(\mathbf{1}-q^*)_j
\end{aligned}$$

If, on the other hand, x_j has $P_j(\mathbf{1}) < 1$, then $x_j = P_j(x)$ has Form$_+$, and, as in the proof of Lemma 5.6, and specifically footnote (8), we must have

$$P_j(\mathbf{1}) \leq 1 - 2^{-|P|} \tag{16}$$

We thus have that:

$$\begin{aligned}
(B(\frac{1}{2}(q^*+y))(\mathbf{1}-q^*))_j &= \sum_{l=1}^{n} p_{j,l}(\mathbf{1}-q^*)_l \\
&= (\sum_{l=1}^{n} p_{j,l}) + p_{j,0} - (\sum_{l=1}^{n} p_{j,l}q_l^*) - p_{j,0} \\
&= P_j(\mathbf{1}) - P_j(q^*) \\
&= P_j(\mathbf{1}) - q_j^* \\
&\leq (1-2^{-|P|}) - q_j^* \qquad \text{(by (16))} \\
&= (\mathbf{1}-q^*)_j - 2^{-|P|} \\
&\leq (1-2^{-|P|})(\mathbf{1}-q^*)_j
\end{aligned}$$

Thus, we can apply Lemma 5.3, by setting $A := B(\frac{1}{2}(y+q^*))$, $u := (\mathbf{1}-q^*)$, $\alpha := 2^{-|P|}$, $\beta := \min\{\frac{1}{2}(1-y)_{\min}, 2^{-|P|}\}$, and we obtain:

$$\|(I - B(\frac{1}{2}(y+q^*)))^{-1}\|_\infty \leq n(\mathbf{1}-q^*)_{\min}^{-2} \max\{2(1-y)_{\min}^{-1}, 2^{|P|}\}2^{|P|}$$

Recall that, by Theorem 3.14, $(\mathbf{1} - q^*)_{\min} \geq 2^{-4|P|}$. Thus

$$\begin{aligned} \|(I - B(\frac{1}{2}(y + q^*)))^{-1}\|_\infty &\leq n2^{9|P|}\max\{2(1-y)_{\min}^{-1}, 2^{|P|}\} \\ &\leq 2^{10|P|}\max\{2(1-y)_{\min}^{-1}, 2^{|P|}\}. \end{aligned}$$

This completes the proof of Part *(i)* of Theorem 5.1.

Proof of Theorem 5.1, Part (ii).

If all variables are of Form$_+$, then Lemma 5.6 gives that, for any $x \in \mathbb{R}^n$, $\|I - B(x)\|_\infty \leq n2^{2|P|}$ and we are done. So assume that there is a variable x_i with $x_i = P_i(x)$ of Form$_*$. Since this part assumes that $x = P(x)$ is strongly connected, every variable depends on it. We quote the following from [24]:

Lemma 5.8 (see proof of Theorem 8.1 in [24])**.** *If $x = P(x)$ is strongly connected and $q^* > \mathbf{0}$, then $q^* = 1$ if and only if $\rho(B(\mathbf{1})) \leq 1$.*

$B(\mathbf{1})$ is a non-negative irreducible matrix. Perron-Frobenius theory gives us that there is a positive eigenvector $v > 0$, with associated eigenvalue $\rho(B(\mathbf{1}))$, the spectral radius of $B(\mathbf{1})$, i.e., such that $B(\mathbf{1})v = \rho(B(\mathbf{1}))v$. But $\rho(B(\mathbf{1})) \leq 1$ so $B(\mathbf{1})v \leq v$.

Lemma 5.9 (cf Lemma 5.9 of [16])**.** $\frac{\|v\|_\infty}{v_{\min}} \leq 2^{|P|}$.

Proof. For any x_i, x_j, there is some $1 \leq k \leq n$ with $(B(1)^k)_{ij} > 0$. We know that $B(1)^k v \leq v$. So $(B(1)^k)_{ij} v_j \leq (B(1)^k v)_i = \rho(B(1))^k v_i \leq v_i$. But by Lemma 5.2 (ii), $(B(1)^k)_{ij} \geq 2^{-|P|}$. So $\frac{v_j}{v_i} \leq 2^{|P|}$. There are v_i,v_j that achieve $v_i = v_{\min}$ and $v_j = \|v\|_\infty$, so we are done. □

We can normalize the top eigenvector, v, so we can assume that $\|v\|_\infty = 1$. Then $v_{\min} \geq 2^{-|P|}$. Consider any equation $x_i = P_i(x) = x_j x_k$ of Form$_*$ (we have already dealt with the case where no such equation exists):

$$\begin{aligned} (B(y)v)_i &= y_j v_k + y_k v_j \\ &\leq y_{\max} v_k + y_{\max} v_j \qquad (\text{where } y_{\max} := \max_r y_r) \\ &= (1 - (\mathbf{1} - y)_{\min})(v_k + v_j) \\ &= (1 - (1 - y)_{\min})(B(\mathbf{1})v)_i \\ &= (1 - (\mathbf{1} - y)_{\min})\rho(B(\mathbf{1}))v_i \\ &\leq (1 - (\mathbf{1} - y)_{\min})v_i \qquad (\text{because } \rho(B(1)) \leq 1) \end{aligned}$$

Now we can apply Lemma 5.3, with $A := B(y)$, $u := v$, $\alpha := 2^{-|P|}$, and $\beta := (\mathbf{1} - y)_{\min}$, to obtain that:

$$\|(I - B(y))^{-1}\|_\infty \leq nv_{\min}^{-2}(\mathbf{1} - y)_{\min}^{-1} 2^{|P|}$$

Inserting our bound for $v_{\min}$, namely $v_{\min} \geq 2^{-|P|}$, yields:

$$\begin{aligned} \|(I - B(y))^{-1}\|_\infty &\leq n2^{3|P|}(\mathbf{1} - y)_{\min}^{-1} \\ &\leq 2^{4|P|}(\mathbf{1} - y)_{\min}^{-1}. \end{aligned}$$

This completes the proof of Part *(ii)* of Theorem 5.1. □

A consequence of Theorem 5.1 and Theorem 3.14 is the following.

Corollary 5.10. *If $x = P(x)$ is a PPS in SNF form with LFP q^*, and $\mathbf{0} < q^* < \mathbf{1}$, then $(I - B(q^*))^{-1}$ exists and is non-negative, and*

$$\|(I - B(q^*))^{-1}\|_\infty \leq 2^{14|P|+1}$$

Proof. Applying part (i) of Theorem 5.1, and letting $y := q^*$, we obtain

$$\begin{aligned} \|(I - B(q^*))^{-1}\|_\infty &\leq 2^{10|P|} \max\{2(\mathbf{1} - q^*)^{-1}_{\min}, 2^{|P|}\} \\ &\leq 2^{10|P|} \max\{2(2^{-4|P|})^{-1}, 2^{|P|}\} \qquad \text{(by Theorem 3.14)} \\ &= 2 \cdot 2^{14|P|} = 2^{14|P|+1}. \end{aligned}$$

□

6 Quadratic convergence for Newton's method on PPSs, and quantitative decision problems for PPSs in unit-cost-P-time

In this section we extend Theorem 3.2 to a *quadratic convergence* result for Newton's method on PPSs, with all constants explicit. Namely, given a PPS, $x = P(x)$, with LFP $\mathbf{0} < q^* < \mathbf{1}$, if we start Newton iteration at $x^{(0)} := \mathbf{0}$, then for all $i \geq 1$, we have

$$\|q^* - x^{(18|P|+2+i)}\|_\infty \leq \frac{1}{2^{2^i}}$$

We then use this result to show that the ***decision problem*** for the LFP q^* of PPSs, which asks, given a PPS $x = P(x)$ over n variables, and given a rational number $r \in [0, 1]$, decide whether $q^*_i > r$ (or whether $q^*_i \geq r$) is in unit-cost P-time, i.e. it is decidable in polynomial time *in the unit-cost arithmetic RAM model of computation*. Combined with the previous hardness result from [24], this implies that decision problem for PPS is complete for unit-cost-P. Hence the problem can be solved in polynomial time in the standard (Turing) model if and only if unit-cost-P = P. Recall that in the unit-cost model, all arithmetic operations on rational numbers cost one unit of time, regardless of how long the numbers are, i.e., how many bits are needed to encode them (their numerator and denominator).

A paradigmatic problem that is solvable in polynomial time in the unit-cost model, is the following problem, called `PosSLP`, which stands for *Positive Straight-Line-Program*: Given an arithmetic circuit over the basis $\{+, -, *\}$ with input 1, or equivalently, a sequence of n instructions of the form $x_i := 1$ or $x_i := x_j \theta x_k$ where $\theta \in \{+, -, *\}$ and $j, k < i$ (this is called a *straight-line program*, abbreviated SLP), decide whether the output of the circuit is positive, i.e. whether the final variable $x_n > 0$. Clearly this problem can be solved in linear time in the unit-cost model by simply performing in order the operations. However, the length (number of bits) of the numbers can grow exponentially (consider for example the SLP $x_1 := 1; x_2 := x_1 + x_1$, followed by $x_i := x_{i-1} * x_{i-1}$ for all $i \geq 3$). It is not known whether `PosSLP` can be solved in P in the standard model. In fact, it was shown in [2] that `PosSLP` is in P iff unit-cost-P = P. In [24] we showed that `PosSLP` reduces in P-time to the decision problem for PPS. We show the converse here, that deciding whether $q^*_i > r$ is P-time reducible to `PosSLP`.

We assume throughout this section, w.l.o.g., that every PPS, $x = P(x)$, is in *simple normal form*, and that the LFP, q^* satisfies $\mathbf{0} < q^* < \mathbf{1}$.

Lemma 6.1. *If $x = P(x)$ is a PPS with n variables in simple normal form (SNF), with LFP $\mathbf{0} < q^* < \mathbf{1}$, then for any $z \in \mathbb{R}^n$ such that $\mathbf{0} \leq z \leq q^*$, then*

$$\|q^* - \mathcal{N}(z)\|_\infty \leq 2^{14|P|+1}\|q^* - z\|_\infty^2$$

Proof. Let us first note that

$$\|\frac{B(q^*) - B(z)}{2}\|_\infty \leq |(q^* - z)\|_\infty \tag{17}$$

This holds because $x = P(x)$ is in SNF form, and thus every equation $x_i = P_i(x)$ is either of the form $x_i = x_j x_k$, or else it is a *linear* (affine) equation, of the form $x_i = \sum_{j=1}^n p_j x_j + p_0$. Now, for every i with a nonlinear equation, i.e., where $P_i(x) \equiv x_j x_k$, the i'th row of the Jacobian matrix $B(x)$, contains exactly two non-zero entries: one is $x_j = \frac{\partial P_i(x)}{\partial x_k}$ and the other is $x_k = \frac{\partial P_i(x)}{\partial x_j}$. Thus, if we define the matrix $A = \frac{B(q^*)-B(z)}{2}$, we must have $\sum_{r=1}^n |A_{i,r}| = \frac{(q_j^* - z_j)+(q_k^* - z_k)}{2} \leq \|q^* - z\|_\infty$. Furthermore, for every i with a *linear* equation, the i'th row of the Jacobian matrix $B(x)$ consists of only constants that do not depend on x, and thus in that case $\sum_{r=1}^n |A_{i,r}| = 0 \leq \|q^* - z\|_\infty$. Thus inequality (17) holds.

Now, using Lemma 3.4, and the equation it gives, namely:

$$q^* - \mathcal{N}(z) = (I - B(z))^{-1}\frac{B(q^*) - B(z)}{2}(q^* - z) \tag{18}$$

and taking norms on both sides of this equation, we have:

$$\begin{aligned}
\|q^* - \mathcal{N}(z)\|_\infty &= \|(I - B(z))^{-1}\frac{B(q^*) - B(z)}{2}(q^* - z)\|_\infty \\
&\leq \|(I - B(z))^{-1}\|_\infty \|\frac{B(q^*) - B(z)}{2}\|_\infty \|(q^* - z)\|_\infty \\
&\leq 2^{14|P|+1}\|\frac{B(q^*) - B(z)}{2}\|_\infty \|(q^* - z)\|_\infty \qquad \text{(by Corollary 5.10)} \\
&\leq 2^{14|P|+1}\|(q^* - z)\|_\infty^2 \qquad \text{(by inequality (17))}
\end{aligned}$$

□

Theorem 6.2. *Let $x = P(x)$ be any PPS in SNF form, with LFP q^*, such that $\mathbf{0} < q^* < \mathbf{1}$. If we start Newton iteration at $x^{(0)} := \mathbf{0}$, with $x^{(k+1)} := \mathcal{N}_P(x^{(k)})$, then for any integer $i \geq 1$ the following inequality holds:*

$$\|q^* - x^{(18|P|+2+i)}\|_\infty \leq \frac{1}{2^{14|P|+1+2^i}} \leq \frac{1}{2^{2^i}} \ .$$

Proof. Lemma 6.1 does not gain us much unless $\|q^* - z\|_\infty \leq \frac{1}{2^{14|P|+1}}$. We need to use our previous linear convergence result until we are close enough for quadratic convergence to kick in. By Theorem 3.2, for $18|P| + 2 = (14|P| + 2) + 4|P|$, we have $\|q^* - x^{(18|P|+2)}\|_\infty \leq \frac{1}{2^{(14|P|+2)}}$ and so

$$2^{14|P|+1}\|q^* - x^{(18|P|+2)}\|_\infty \leq \frac{1}{2} \tag{19}$$

Lemma 6.1 tell us that:

$$\|q^* - \mathcal{N}(z)\|_\infty \leq 2^{14|P|+1}\|q^* - z\|_\infty^2$$

Multiplying both sides by $2^{14|P|+1}$ gives:

$$2^{14|P|+1}\|q^* - \mathcal{N}(z)\|_\infty \leq (2^{14|P|+1}\|q^* - z\|_\infty)^2$$

By induction, for any integers $i \geq 0$, $m \geq 0$

$$2^{14|P|+1}\|q^* - x^{(i+m)}\|_\infty \leq (2^{14|P|+1}\|q^* - x^{(m)}\|_\infty)^{2^i}$$

Taking $m = 18|P| + 2$, we can use equation (19):

$$2^{14|P|+1}\|q^* - x^{(18|P|+2+i)}\|_\infty \leq (\frac{1}{2})^{2^i}$$

and so

$$\|q^* - x^{(18|P|+2+i)}\|_\infty \leq \frac{1}{2^{(14|P|+1+2^i)}}$$

□

We next wish to use Theorem 6.2 in order to establish that, we can decide, given a rational number r, whether $q_i^* \geq r$, in time polynomial in $|P|$ and the encoding size of r in the unit-cost model, using Newton's method with *exact* arithmetic. To do this, we need to first establish a separation bound relating to q^* and a given rational r.

Lemma 6.3. *Given a PPS, $x = P(x)$, with n variables, and with LFP q^*, such that $\mathbf{0} < q^* < \mathbf{1}$, and given any rational number $r > 0$, where $r = \frac{a}{b} < 1$ is represented as the ratio of positive integers a and b, with $a \leq b$, then for any $k \in \{1, \ldots, n\}$, if $q_k^* \neq r$, then*

$$|q_k^* - r| \geq 2^{-2(n+1)(\max\{|P|,\log(b)\}+2(n+1)\log(2n+2))5^n}$$

Proof. We shall use the following theorem by Hansen et. al. [34] regarding explicit separation bounds for isolated real-valued solutions to polynomial systems of equations:

Lemma 6.4. *(Theorem 23 from [34]) Consider a polynomial system of equations*

$$(\Sigma) \qquad g_1(x_1, \ldots, x_n) = \cdots = g_m(x_1, \ldots, x_n) = 0 \ , \tag{20}$$

with polynomials of degree at most d and integer coefficients of magnitude at most 2^τ.

If $\gamma_j = (\gamma_{j,1}, \cdots, \gamma_{j,n})$ is an isolated (in Euclidean topology) real solution of (Σ), then for any i, either

$$2^{-2n(\tau+2n\log(dm))(2d+1)^{n-1}} < |\gamma_{j,i}| \quad \text{or} \quad \gamma_{j,i} = 0 \ . \tag{21}$$

To apply Lemma 6.4, we need the fact that $q^* > \mathbf{0}$ is an isolated solution of the PPS. This follows immediately from a more general *unique fixed point* theorem established in [22] (Theorem 18 of [22]) for the equations corresponding to the termination probabilities of general recursive Markov chains (RMCs), and it also follows from (variants of) older results about multi-type branching processes (see [35], Thm. II.7.2 and Corollary II.7.2.1). Specifically, the unique fixed point theorem of [22] establishes that, in particular, if a PPS has LFP q^* with $\mathbf{0} < q^* < \mathbf{1}$, then q^* is the unique solution of $x = P(x)$ in the interior of $[0, 1]^n$, i.e., in $(0, 1)^n$. Thus, it is clearly an isolated solution.

For each x_i, let d_i be the product of the denominators of all coefficients of $P_i(x)$. Then $d_i x = d_i P_i(x)$ clearly has integer coefficients which are no larger than $2^{|P|}$. Also, consider a new variable

y, and a new equation $y = x_k - r$, where $r = \frac{a}{b}$ is the given positive rational value. This equation is clearly equivalent to $by = bx_k - a$. Suppose the PPS, $x = P(x)$, has LFP $q^* > \mathbf{0}$, and for any $k \in \{1, \ldots, n\}$, consider the system of $n+1$ polynomial equations, in $n+1$ variables (with an additional variable y), given by:

$$d_i x_i = d_i P_i(x) \text{ , for all } i \in \{1, \ldots, n\}; \quad \text{and} \quad by = bx_k - a \ . \tag{22}$$

Since $\mathbf{0} < q^* < \mathbf{1}$, we know from the unique fixed point theorem of [22] that q^* is an isolated solution of $x = P(x)$. If $z \in \mathbb{R}^n$ is any solution vector for $x = P(x)$, there is a unique $w \in \mathbb{R}$ such that $x := z$ and $y := w$ forms a solution to the equations (22); namely let $w = z_k - r$. So, letting $x := q^*$, and letting $y := q^*_k - r$, gives us an isolated solution of the equations (22). We can now apply Lemma 6.4 to the system (22). Since $y := q^*_k - r$, equation (21) in Lemma 6.4 says that

$$2^{-2(n+1)(\max\{|P|,\log(b)\}+2(n+1)\log(2n+2))5^n} < |q^*_k - r| \ , \quad \text{or else} \quad q^*_k - r = 0 \ .$$

which is just what we wanted to establish. □

We are now ready to establish the following:

Theorem 6.5. *Given a PPS, $x = P(x)$, with n variables, and with LFP $\mathbf{0} < q^* < \mathbf{1}$, and given a rational number $r = a/b \in (0, 1]$, where a and b are positive integers given in binary. Let $g = 32|P| + 4 + 6n + 56(\lceil \log(n) \rceil + \lceil \log(|P|) \rceil + \lceil \log(\log b) \rceil)$. Let $x^{(i)}$ denote the i'th Newton iterate starting at $x^{(0)} := \mathbf{0}$, applied to the PPS $x = P(x)$. Let $m := 2 + 3n + 28(\lceil \log(n) \rceil + \lceil \log(|P|) \rceil + \lceil \log(\log b) \rceil)$. Then for any $k \in \{1, \ldots, n\}$,*

1. *$q^*_k > r$ if and only if $x^{(g)}_k > r$.*
2. *$q^*_k < r$ if and only if $x^{(g)}_k + 2 \cdot \frac{1}{2^{2^m}} < r$.*

Proof. Let $\gamma = 2^{-2(n+1)(\max\{|P|,\log(b)\}+2(n+1)\log(2n+2))5^n}$. Recall that Lemma 6.3 tells us that $|q^*_k - r| \geq \gamma$, for any k, unless $q^*_k = r$. We know $x^{(g)} \leq q^*$. Furthermore, g has been chosen so that, by Theorem 6.2, $\|q^* - x^{(g)}\|_\infty < \frac{1}{2^{2^m}} < \gamma/8$.

To establish (1.), in one direction we simply note that if $x^{(g)}_k > r$, then since $q^*_k > x^{(g)}_k$, we must have $q^*_k > r$. In the other direction, if $q^*_k > r$, then $q^*_k - r \geq \gamma$, but we know $q^*_k - x^{(g)}_k \leq \gamma/8$, so $x^{(g)}_k \geq r + \frac{7}{8}\gamma \geq r$.

To establish (2.), in one direction since $q^*_k - x^{(g)}_k < \frac{1}{2^{2^m}}$, we have $q^*_k < x^{(g)}_k + 2 \cdot \frac{1}{2^{2^m}}$, and thus if $x^{(g)}_k + 2 \cdot \frac{1}{2^{2^m}} < r$, then $q^*_k < r$. In the other direction, if $q^*_k < r$, then since $r - q^*_k \geq \gamma$, and since $q^*_k \geq x^{(g)}_k$, and since $2 \cdot \frac{1}{2^{2^m}} \geq \gamma/4$, we have $x^{(g)}_k + 2 \cdot \frac{1}{2^{2^m}} \geq \gamma/4 < r$. This completes the proof. □

Corollary 6.6. *Given a PPS, $x = P(x)$, with n variables, and with LFP $q^* \in [0, 1]^n$, given a coordinate $k \in \{1, \ldots, n\}$, and given a rational number $r \in [0, 1]$, there is an algorithm that determines which of the following cases holds: (A) $q^*_k < r$, or (B) $q^*_k = r$, or (C) $q^*_k > r$.*
The algorithm runs in time polynomial in $|P|$, the bit encoding size of the PPS, and size(r), the binary encoding size of r, in the unit-cost arithmetic RAM model of computation.

Hence the problems of deciding whether $q^*_k > r$, or deciding whether $q^*_k < r$, are complete for unit-cost-P. Furthermore, both these problems are P-time many-one (Karp) reducible to `PosSLP`*, and hence they are P-time equivalent to* `PosSLP`*.*

The same results hold for the extinction probability of a multitype branching process, the probability of the language of a SCFG, and the termination probability of a 1-RMC.

Proof. First, we note that deciding whether $q_k^* = 0$ and whether $q_k^* = 1$, can be carried out in P-time ([24]). Hence, we can detect and remove in P-time all variables x_i such that $q_i^* \in \{0, 1\}$. Then we are left with a residual PPS, $x = P(x)$, with LFP q^* such that $\mathbf{0} < q^* < \mathbf{1}$.

Notice that each iteration of Newton's method, $x^{(j+1)} = \mathcal{N}(x^{(j)}) = x^{(j)} + (I - B(x^{(j)}))^{-1}(P(x^{(j)}) - x^{(j)})$, on a PPS, $x = P(x)$ with n variables, can be computed by performing a $n \times n$ matrix inversion and matrix-vector multiplication and summing of vectors. Each matrix inversion can be done with $O(n^3)$ arithmetic operations, matrix-vector multiplication takes $O(n^2)$ operations and vector summation $O(n)$ operations. Thus, each Newton iteration uses $O(n^3)$ operations.

Now we apply Theorem 6.5. Since the number of iterations g given in the statement of Theorem 6.5 is polynomial in $|P|$ and $size(r)$ (in fact, even in $\log(size(r))$) we can compute $x^{(g)}$ in polynomial time in the unit-cost arithmetic RAM model of computation. Likewise, since the m given in the statement of the Theorem is also polynomial in $|P|$ and $size(r)$, we can use repeated squaring to compute $\frac{1}{2^{2^m}}$ in time polynomial in $|P|$ and $size(r)$ (i.e., with polynomially many arithmetic operations). We can also add two numbers at unit-cost to obtain $x_k^{(g)} + 2 \cdot \frac{1}{2^{2^m}}$.

In order to determine whether $q_k^* > r$, we simply need to check whether $x_k^{(g)} > r$, and to determine whether $q_k^* < r$ we simply need to check whether $x_k^{(g)} + 2 \cdot \frac{1}{2^{2^m}} < r$.

This shows that the decision problems for a PPS are in unit-cost-P. From the hardness result in [24], it follows that they are complete for unit-cost-P.

Allender et. al. showed in [2] that every discrete decision problem (with rational valued inputs) that can be decided in P-time in the unit-cost arithmetic RAM model of computation is in $\mathbf{P}^{\texttt{PosSLP}}$, i.e. can be solved in polynomial time in the standard model provided one is given a subroutine for `PosSLP` that runs in P (in the standard model). This type of reduction allows the algorithm to make many calls to the subroutine. We show now that the decision problems for a PPS, i.e., deciding whether $q_k^* > r$, and deciding whether $q_k^* < r$, are in fact P-time *many-one* (Karp) reducible to `PosSLP`, i.e. that one can construct from a given PPS and rational r in P-time an instance of `PosSLP` whose answer gives the answer to the decision problem for the PPS.

If $r \in \{0, 1\}$, we have already pointed out that deciding both $q_k^* > r$ and $q_k^* < r$ is solvable in (strongly) polynomial time ([24, 15]), thus there is nothing to prove in this case. So, suppose $r \in (0, 1)$, and suppose that $\mathbf{0} < q^* < \mathbf{1}$. First note that we can easily construct arithmetic circuits over $\{+, -, *, /\}$ with input 1 of polynomial size that compute the coefficients that appear in the polynomials of the PPS. For each Newton iteration j, we can construct an arithmetic circuit C_j which takes as input $x^{(j-1)}$ and produces as output $x^{(j)}$. Regarding the matrix inversion, note that each entry can be expressed using Cramer's rule as the ratio of matrix determinants, and it is well known how to construct a circuit with polynomially many gates for a determinant. Since $\mathbf{0} < q^* < \mathbf{1}$, we have established in Theorem 6.5 that $q_k^* > r$ if and only if $x_k^{(g)} > r$, and likewise that $q_k^* < r$ if and only if $x_k^{(g)} + 2 \cdot \frac{1}{2^{2^m}} < r$. Concatenating all the circuits C_j for $j = 1$ to g and feeding 0=1-1 into C_1 yields a circuit C that computes $x^{(g)}$, and has size polynomial in $|P|$ and $size(r)$. Therefore, we can also obtain a circuit C' of size polynomial in $|P|$ and $size(r)$ that computes $x_k^{(g)} + 2 \cdot \frac{1}{2^{2^m}}$, for any desired coordinate k, because m is polynomial in $|P|$ and $size(r)$, and we can use a repeated squaring circuit to compute $\frac{1}{2^{2^m}}$.

As shown in [2] (see also [24]), division gates in arithmetic circuits over $\{+, -, *, /\}$ can be removed by keeping track of numerators and denominators separately. Thus, the numerator n_1 and

denominator d_1 of the rational coordinate $x_k^{(g)}$ of the vector $x^{(g)}$ (as well as the numerator n_1' and denominator d_1' of the rational value $x_k^{(g)} + 2 \cdot \frac{1}{2^{2^m}}$) can be computed by polynomial-sized arithmetic circuits over $\{+, -, *\}$ which can be constructed in P-time given $x = P(x)$ and r. Obviously the rational number r can also have its numerator n_2 and denominator d_2 represented this way by circuits in P-time. Consequently, to decide whether $x_k^{(g)} > r$ (likewise, whether $x_k^{(g)} + 2 \cdot \frac{1}{2^{2^m}} < r$), we combine these four circuits into a circuit that computes $n_1 \cdot d_2 - n_2 \cdot d_1$ (respectively, $n_2 \cdot d_1' - n_1' \cdot d_2$) and test if its output is positive. But `PosSLP` is precisely this problem, so this yields a P-time many-one reduction from both these problems to `PosSLP`. □

7 Conclusions

We have shown that one can approximate the least fixed point solution of a probabilistic polynomial system of equations (a PPS) to any desired accuracy 2^{-k} in time polynomial in the encoding size of the PPS and the number k of bits of accuracy. This result applies in particular to the computation of extinction probabilities of multitype branching processes, the termination probabilities (a.k.a., partition function) of SCFGs and of 1-RMCs. We showed furthermore, that the decision problem of comparing these probabilities with a given rational bound can be solved in polynomial time in the unit-cost arithmetic RAM model of computation.

We mention briefly some related subsequent work. Since the publication of the conference version of this paper [19][9], we have obtained several subsequent results in papers that build on and extend this work. Specifically, in [18] and [20] we have studied more general *branching Markov decision processes*, which extend BPs with a controller that can take actions to influence the reproduction probabilities with the goal of maximizing or minimizing the extinction probability, and we studied also their associated *max/min probabilistic polynomial systems of equations* (max/minPPSs) which involve also a max or min operator. We have shown that a Generalized Newton's method, which involves linear programming in each iteration, can be used to compute both the least and greatest fixed point solutions of such systems of equations in polynomial time to desired precision (in the standard Turing model of computation).

Moreover, in [49] we have studied the behavior of Newton's method on *arbitrary* monotone polynomial systems (MPS) of equations, $x = P(x)$, and we have given worst case upper bounds on the convergence rate of Newton's method, which we have shown are essentially optimal in a number of important parameters of the problem. These upper bounds for general MPSs are however necessarily exponential as a function of the encoding size of $x = P(x)$, and are thus substantially worse than what we have established in this paper for the special cases of PPSs (after suitable qualitative preprocessing).

We finally remark on a connection between computing the LFP of a MPS, $x = P(x)$, and a class of mathematical optimization problems referred to as *geometric programming* (see [13]). Recall, as mentioned in the introduction, that computing the LFP of any MPS, $x = P(x)$, can be formulated as the following optimization problem: *minimize:* $\sum_{i=1}^{n} x_i$; *subject to:* $\{P(x) \leq x;\ x \geq 0\}$. After removing the variables whose LFP value equals 0 from the system $x = P(x)$, which we can do in P-time using simple and-or graph reachability ([24]), the mathematical program described for the LFP

[9]The conference version also included a number of results about stochastic context-free grammars (SCFGs) going beyond computation of their termination probabilities (i.e., their partition function), which will be submitted elsewhere in journal form.

of the MPS can be reformulated as a *geometric programming (GP) problem in posynomial form.*[10] A GP with n variables $x = (x_1, \ldots, x_n)$, involves minimizing a posynomial function $f_0(x)$, subject to posynomial constraints $f_j(x) \leq 1$, $j = 1, \ldots, m$, and $x > 0$ (i.e., all variables are constrained to be strictly positive). A posynomial, $f_i(x)$, is a positive-weighted sum of generalized monomials, where a generalized monomial has the form $x_1^{v_1} x_2^{v_2} \ldots x_n^{v_n}$, with each v_i a rational number (not necessarily positive). Note that the original objective $\sum_{i=1}^{n} x_i$ is clearly a posynomial, and the constraints $P_i(x) \leq x_i$ can be reformulated as posynomial constraints $\frac{P_i(x)}{x_i} \leq 1$, by diving by x_i, for any MPS $x = P(x)$. After removing the variables that are 0 in the LFP via preprocessing, we can add the constraint $x > 0$, requiring all remaining variables to be positive, thus yielding a GP in posynomial form. Geometric programs in posynomial form are not convex programs, but they can be transformed into convex programs by using a change of variables, $y_i = \log x_i$ (see [13]). Established methods exist for tackling geometric programs, based on using this transformation and applying convex optimization methods (see, e.g., [10]). However, it is important to note that in general computing a non-trivial approximation of an optimal feasible solution for a GP, or even just a non-trivial approximation of the optimal value (even when the program is guaranteed to be feasible, and even when the feasible region is guaranteed to be bounded in, say, $[0, 1]^n$), cannot be done in polynomial time (in the standard Turing model) unless `PosSLP` can be solved in polynomial time, i.e., unless the unit-cost exact arithmetic model can be simulated by the Turing machine model with only polynomial overhead. This is a consequence of a result in [24] (Theorem 5.2), which established that obtaining *any* nontrivial approximation of the termination probabilities of a RMC is `PosSLP`-hard, and the fact ([24]) that we can easily in P-time construct from any RMC a corresponding MPS whose LFP is the vector of termination probabilities. Thus we can also construct a feasible GP in posynomial form, whose feasible set is in $[0, 1]^n$, and whose unique optimal solution yields the vector of termination probabilities of the RMC. It is nevertheless interesting to ask whether, for the special case of PPSs studied in this paper, we can, after some polynomial-time preprocessing, use geometric programming and convex optimization to obtain an alternative P-time algorithm for approximating the LFP of a PPS. It turns out that we can show, by exploiting in detail the results established in this paper for the analysis of Newton's method on PPSs, that sufficient conditions do hold for the Ellipsoid method to be applicable in P-time (see [27]) on the log-transformed convex GPs corresponding to suitably preprocessed PPSs, where the variables that equal 0 or 1 in the LFP have been removed. (We do not provide the rather technical derivations of this fact in this paper.) This leads to an alternative P-time algorithm for approximating the LFP of a PPS based on the Ellipsoid method, although not a practical one. (In fact, we can show more generally that GPs can be used, after suitable preprocessing of maxPPSs, to approximate the LFP of maxPPSs (but not minPPSs) in P-time. As mentioned earlier, the first P-time algorithms for approximating the LFP of max/minPPSs was given in our subsequent work [18], based on a generalized Newton's method.)

References

[1] S. P. Abney, D. A. McAllester, and F. Pereira. Relating probabilistic grammars and automata. In *Proc. of 27th Meeting of the Assocation for Computational Linguistics (ACL'99)*, pages 542–549, 1999.

[10] We thank Pablo Parillo for pointing out to us this connection to geometric programming.

[2] E. Allender, P. Bürgisser, J. Kjeldgaard-Pedersen, and P. B. Miltersen. On the complexity of numerical analysis. *SIAM J. Comput.*, 38(5):1987–2006, 2009.

[3] T. Apostol. *Mathematical Analysis*. Addison-Wesley, 2nd edition, 1974.

[4] K. B. Athreya and P. E. Ney. *Branching Processes*. Springer-Verlag, 1972.

[5] J. K. Baker. Trainable grammars for speech recognition. In *Proc. of Spring Conference of the Acoustical Society of America*, pages 547–550, 1979.

[6] N. G. Bean, N. Kontoleon, and P.G. Taylor. Markovian trees: properties and algorithms. *Annals of Operations Research*, 160(1):31-50, 2008.

[7] A. Berman and R. J. Plemmons. *Nonnegative Matrices in the Mathematical Sciences*. Classics in Applied Mathematics. SIAM, 1994.

[8] D. A. Bini, G. Latouche, and B. Meini. Solving non-linear matrix equations arising in tree-like stochastic processes. *Linear Algebra and its Applications*, 366:39–64, 2003.

[9] D. Bini, G. Latouche, and B. Meini. *Numerical methods for Structured Markov Chains*. Oxford University Press, 2005.

[10] S. Boyd, S. J. Kim, L. Vandenberghe, and A. Hassibi. A tutorial on geometric programming. Optim. Eng., 8(1):67-127, 2007.

[11] T. Brázdil, J. Esparza, and A. Kučera. Analysis and prediction of the long-run behavior of probabilistic sequential programs with recursion. In *Proc. FOCS*, pages 521–530, 2005.

[12] P. Cameron. *Combinatorics: topics, techniques, algorithms*. Cambridge U. Press, 1994.

[13] R. J. Duffin, E. L. Peterson, and C. Zener. *Geometric Programming*. John Wiley and Sons, 1967.

[14] R. Durbin, S. R. Eddy, A. Krogh, and G. Mitchison. *Biological Sequence Analysis: Probabilistic models of Proteins and Nucleic Acids*. Cambridge U. Press, 1999.

[15] J. Esparza, A. Gaiser, and S. Kiefer. Computing least fixed points of probabilistic systems of polynomials. In *Proc. 27th STACS*, pages 359–370, 2010.

[16] J. Esparza, S. Kiefer, and M. Luttenberger. Computing the least fixed point of positive polynomial systems. *SIAM Journal on Computing*, 39(6):2282–2355, 2010.

[17] J. Esparza, A. Kučera, and R. Mayr. Model checking probabilistic pushdown automata. *Logical Methods in Computer Science*, 2(1):1 – 31, 2006.

[18] K. Etessami, A. Stewart, and M. Yannakakis. Polynomial-time algorithms for Branching Markov Decision Processes, and probabilistic min(max) polynomial Bellman equations. In *Proc. 39th Int. Coll. on Automata, Languages and Programming (ICALP)*, 2012. (See full preprint Arxiv:1202.4789, available at: `http://arxiv.org/abs/1202.4798` .)

[19] K. Etessami, A. Stewart, and M. Yannakakis. Polynomial-time algorithms for multi-type branching processes and stochastic context-free grammars. In *Proc. 44th ACM Symposium on Theory of Computing (STOC)*, 2012.

[20] K. Etessami, A. Stewart, and M. Yannakakis. Greatest Fixed-Points of probabilistic min/max polynomial equations, and reachability for branching Markov decision processes. In *Proc. 42nd Int. Coll. on Automata, Languages and Programming (ICALP)*, 2015. (See full preprint Arxiv:1502.05533, available at: `http://arxiv.org/abs/1502.05533` .)

[21] K. Etessami, A. Stewart, and M. Yannakakis. A note on the complexity of geometric programming and analysis of stochastic processes. *forthcoming*, 2016.

[22] K. Etessami and M. Yannakakis. Model checking of recursive probabilistic systems. *ACM Transactions on Computational Logic*, 13(2):12, 2012.

[23] K. Etessami, D. Wojtczak, and M. Yannakakis. Quasi-birth-death processes, tree-like QBDs, probabilistic 1-counter automata, and pushdown systems. *Perform. Eval.*, 67(9):837–857, 2010. (Conference version in *Proc of 5th International Conference on the Quantitative Evaluation of Systems (QEST)*, pages 243–253, 2008.)

[24] K. Etessami and M. Yannakakis. Recursive Markov chains, stochastic grammars, and monotone systems of nonlinear equations. *Journal of the ACM*, 56(1), 2009. (Conference version appeared in STACS'2005.)

[25] K. Etessami and M. Yannakakis. On the complexity of Nash equilibria and other fixed points. *SIAM Journal on Computing*, 39(6):2531–2597, 2010.

[26] R. Fagin, A. Karlin, J. Kleinberg, P. Raghavan, S. Rajagopalan, R. Rubinfeld, M. Sudan, and A. Tomkins. Random walks with "back buttons". *Annals of Applied Probability*, 11(3):810–862, 2001.

[27] M. Grötschel, L. Lovasz, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer-Verlag, 1993.

[28] P. Haccou, P. Jagers, and V. A. Vatutin. *Branching Processes: Variation, Growth, and Extinction of Populations*. Cambridge U. Press, 2005.

[29] S. Hautphenne. Extinction probabilities of supercritical decomposable branching processes. J. Appl. Prob., volume 49, pages 639-651, 2012.

[30] S. Hautphenne. A structured Markov chain approach to branching processes. *Stochastic Models*, volume 31, pages 403-432, 2015.

[31] S. Hautphenne, G. Latouche, and M. A. Remiche. Newton's iteration for the extinction probability of a Markovian binary tree. *Linear Algebra Appl.*, 428(11-12): 27912804, 2008.

[32] S. Hautphenne, G. Latouche, and M. A. Remiche. Algorithmic approach to the extinction probability of branching processes. *Methodol. Comput. Appl. Probab.*, 13(1): 171192, 2011.

[33] S. Hautphenne and B. Van Houdt. On the link between Markovian trees and tree-structured Markov chains. *European J. Oper. Res.*, 201(3): 791798, 2010.

[34] K. Arnsfelt Hansen, M. Koucký, N. Lauritzen, P. Bro Miltersen, and E. P. Tsigaridas. Exact algorithms for solving stochastic games: extended abstract. In *STOC*, pages 205–214, 2011. see full Arxiv version, arXiv:1202.3898 (2012).

[35] T. E. Harris. *The Theory of Branching Processes*. Springer-Verlag, 1963.

[36] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, 1979.

[37] R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 1985.

[38] E. Isaacson and H. B. Keller. *Analysis of Numerical Methods*. J. Wiley & Sons, 1966.

[39] M. Kimmel and D. E. Axelrod. *Branching processes in biology*. Springer, 2002.

[40] A. N. Kolmogorov and B. A. Sevastyanov. The calculation of final probabilities for branching random processes. *Doklady*, 56:783–786, 1947. (Russian).

[41] K. Lari and S. J. Young. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech & Language*, 4(1):35 – 56, 1990.

[42] G. Latouche and V. Ramaswami. *Introduction to Matrix Analytic Methods in Stochastic Modeling*. ASA-SIAM series on statistics and applied probability, 1999.

[43] C. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.

[44] M.-J. Nederhof and G. Satta. Computing partition functions of PCFGs. *Research on Language and Computation*, 6(2):139–162, 2008.

[45] M.-J. Nederhof and G. Satta. Probabilistic parsing. *New Developments in Formal Languages and Applications*, 113:229–258, 2008.

[46] M. F. Neuts. *Matrix-Geometric Solutions in Stochastic Models: an algorithmic approach*. Johns Hopkins University Press, 1981.

[47] J. M. Ortega and W.C. Rheinbolt. *Iterative solution of nonlinear equations in several variables*. Academic Press, 1970.

[48] Y. Sakakibara, M. Brown, R Hughey, I.S. Mian and̃ K. Sjolander, R. Underwood, and D. Haussler. Stochastic context-free grammars for tRNA modeling. *Nucleic Acids Research*, 22(23):5112–5120, 1994.

[49] A. Stewart, K. Etessami, and M. Yannakakis. Upper bounds for Newton's method on monotone polynomial systems, and P-Time model checking of probabilistic one-counter automata. *Journal of the ACM*, 62(4), 2015. Conference version appeared in *Proc. 25th Int. Conf. on Computer Aided Verification*, 2013.

[50] D. Wojtczak and K. Etessami. Premo: an analyzer for probabilistic recursive models. In *Proc. 13th Int. Conf. on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, pages 66–71, 2007.